GENERATIVE AND DISCRIMINATIVE MODELS FOR PERSON VERIFICATION AND EFFICIENT SEARCH

BY

ZHEN LI

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2013

Urbana, Illinois

Doctoral Committee:

Professor Thomas S. Huang, Chair
Professor Mark Hasegawa-Johnson
Associate Professor Feng Liang
Professor Zhi-Pei Liang

ABSTRACT

This dissertation studies the person verification problem in modern surveillance and video retrieval systems. The problem is to identify whether a pair of face or human body images is about the same person, even if the person is not seen before. Traditional methods either model the intrapersonal and extrapersonal variations with probabilistic distributions, or look for a distance (or similarity) measure between images (e.g., by metric learning algorithms), and make decisions based on a fixed threshold. We show that the resulting decisions, depending merely on pairwise image differences, are nevertheless insufficient and sub-optimal for the verification problem. In this dissertation, we study both generative and discriminative models for person verification. Both methods consider a joint model of two images in a pair, and provide a decision function of second-order form that generalizes from previous approaches. We also generalize our model to a multi-setting scenario, where environment mismatch, a major challenge in cross-setting person verification, is handled. We evaluate our algorithms on face verification and human body verification problems on a number benchmark datasets, such as Multi-PIE, LFW, CIGIT-AIS, VIPer, VIPeR, and CAVIAR4REID. Our methods outperform not only the classical Bayesian Face Recognition approach, metric learning algorithms (LMNN, ITML, etc.), but also the state-of-the-art in the computer vision community.

This dissertation also considers efficient person search, a potential application of person verification in surveillance systems. To this end, we propose a general learning-to-search framework for efficient similarity search in high dimensions. Experimental results show that our approach significantly outperforms the state-of-the-art learning-to-hash methods (such as spectral hashing), as well as state-of-the-art high-dimensional search algorithms (such as LSH and k-means trees).

To my family.

ACKNOWLEDGMENTS

I would like to express my sincerest gratitude to my adviser, Prof. Thomas S. Huang, for his continuous supervision, support, encouraging and care, without which this dissertation would not have been possible. I am grateful to Prof. Mark Hasegawa-Johnson, Prof. Feng Liang, and Prof. Zhi-Pei Liang, for serving on my doctoral committee, and offering invaluable comments and suggestions in the preparation of this dissertation. I thank all my collaborators: Tong Zhang, Yihong Gong, Yun Fu, Xi Zhou, Liangliang Cao, Huazhong Ning, Shiyu Chang, Minqiang Xu. It is my great honor to work with and learn from them. Thanks also go to many other friends, in particular former and current members of the Image Formation and Processing (IFP) group. It is their support and care that makes this journey truly exciting and enjoyable. Finally, I owe my parents and my wife Ying huge appreciation for their love, understanding, and support throughout the years. This dissertation is dedicated to them.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Person verification – "Are you the person you claim to be?" – is an important problem with many applications. Modern image retrieval systems often want to verify whether photos contain the same person or the same object. Person verification also gets more and more important for social network websites, where it is highly preferred to automatically assign person tags to user uploaded photos. More importantly, the huge amount of surveillance cameras – there are more than 30 million surveillance cameras in U. S. recording about 4 billion hours of videos per week – calls for reliable systems which are able to identify the same person across different videos, a critical task that cannot merely rely on human labors. So developing an automatic verification system is of great interest in practice.

There are two main visual clues for person verification: face images and human body figures. Although our human vision system has the amazing ability of performing verification – we can judge whether two images are about the same person without even seeing that person before – it is difficult to build a computer-based automatic system for this purpose. The two major challenges are:

- **Limited-sample issue.** For a given query image, the person in the image might have never appeared before, or has only one or very few images available in the database. This makes it impossible to train a person-dependent model, which is usually the strategy taken in traditional classification problems.

- **Environment mismatch.** The query image and the other images in the database are rarely collected in exactly the same environment. The huge intra-person variations including viewpoint, lighting condition, image quality, resolution, etc., pose tremendous challenges to building robust person verification systems.

Figure 1.1 provides some examples illustrating the difficulties with the person verification problem.

Figure 1.1: Example images of George W. Bush showing huge intra-person variations.

The problem of person verification arises from a more general and better-known problem - person identification (or recognition). Generally speaking, person identification refers to assigning a query image to one of the several known identities, each of which has a number of images available in the database.[1] With images represented as numerical feature vectors, the person identification problem becomes a standard classification problem. Then it can be solved by either generative models that characterize the person-dependent feature distributions, or discriminative classifiers such as Support Vector Machines (SVMs) [3].

The key difference between person identification and person verification – a fundamental and more practical case of the former problem – is the limited-sample issue we describe above. And due to this, traditional classification methods would no longer apply. Specifically, in person verification, each person usually has only one or very few images in the database (imagine a surveillance scenario where only a few ID photos are available for target subjects), which makes it intractable to either reliably estimate the class-conditional distributions or learn robust discriminative classifiers. Therefore, the entire problem needs to be reconsidered from the machine learning perspective.

We now define the person verification problem under the setting of "identification with limited samples," or in an extreme case, "identification with one sample per person." This naturally implies a more fundamental problem - to identify

[1] In some cases, the system is also required to determinate if the query image is not from any of the known people, or in other words, is from an imposter.

whether two images are about the same person, which we formally refer to as the **person verification** problem. Mathematically, it can be described as follows: for a pair of images represented by x, $y \in \mathbb{R}^d$, respectively, each of which corresponds to an identity label $m(x)$ and $m(y)$, we aim to decide whether they are from the same person, i.e., $m(x) = m(y)$, or not.

We should note that the verification problem is fundamentally different from learning a classifier traditional machine learning problems. Traditional machine learning algorithms consider individual samples instead of a pair of samples. This paired setup for verification naturally imposes some symmetry constraint between x and y, a constraint seldom seen in ordinary learning algorithms. More importantly, for verification, of interest is to determine whether a pair of samples is from the same category or not, but not to answer which category/categories they belong to.

This dissertation focuses on machine learning aspects of the verification problem. The general methodology for person verification is shown in Figure 1.2. Offline we form a training set by collecting many images from many people. The training set, with groundtruth identity labels, is used to learn a *verification model* that takes as input a pair of images. In the testing phase, we apply the learned model to a pair of images from some unseen people, and determine if these two images are from the same person.

In principle, the verification models fall into two categories: generative and discriminative.

- Based on Bayesian decision theory [4], the verification problem can be solved by computing the log-likelihood ratio (LLR) of two hypothesis testings:

$$LLR(x, y) = \log \frac{p(x, y|\mathcal{H}_s)}{p(x, y|\mathcal{H}_d)}, \tag{1.1}$$

  where $\mathcal{H}_s$ and $\mathcal{H}_d$ are:

  - $\mathcal{H}_s$: image x and y are from the same person;
  - $\mathcal{H}_d$: image x and y are from different people.

- More generally, we could directly learn a **decision function**, $f(x, y) \approx$

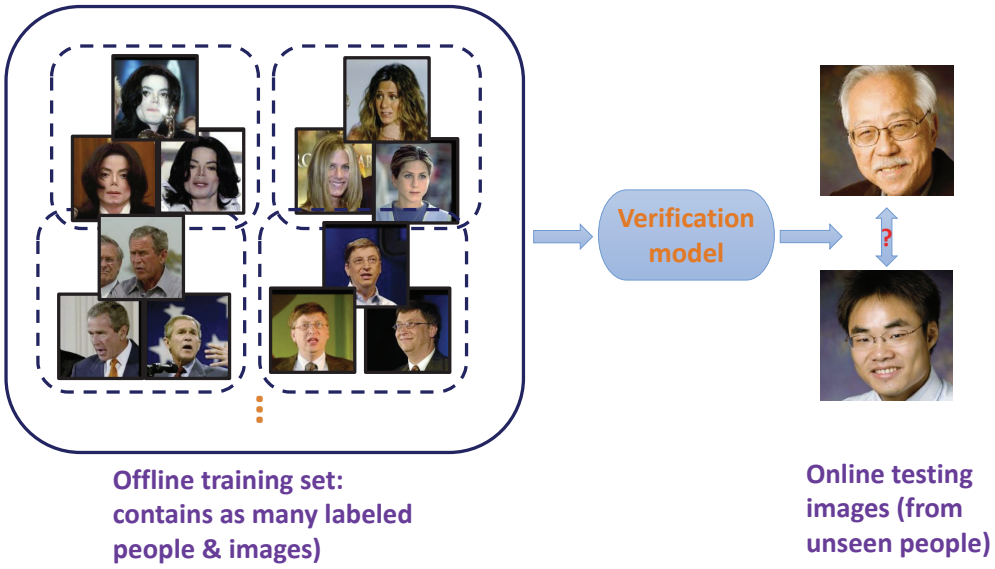


Figure 1.2: General methodology for person verification.

$LLR(x, y)$, with the following properties in its ideal case:

$$f(x,y) \begin{cases} > 0, & \text{if } m(x) = m(y); \\ < 0, & \text{otherwise.} \end{cases} \tag{1.2}$$

For either type of method, an appropriate model family or function family needs to be presumed. Then we learn the model parameters on the training set under certain criterion (e.g. maximum likelihood). In the testing phase, we make the decision, whether two images are about the same person, by evaluating either the log-likelihood ratio in (1.1) or the decision function in (1.2), and comparing against certain threshold.[2] It should be noted that both $LLR(x, y)$ and $f(x, y)$ must be symmetric with respect to x and y.

The rest of the dissertation is organized as follows. In Chapter 2, we first discuss the relationship between person verification and identification, and briefly review some of the previous works on person verification. In Chapter 3 we propose a generative model for the general person verification problem following (1.1). A discriminative approach that directly learns the decision function in (1.2) is in-

[2]The threshold is usually set to be 0, if the prior probabilities of the two hypotheses are regarded as being equal. In practice, one might need to tune the threshold in order to meet certain operating requirements (e.g. false alarm rate below 1%).

vestigated in Chapter 4. In Chapter 5 we generalize our model to a multi-setting scenario, where environment mismatch is handled for cross-setting person verification. Chapter 6 studies person search, a potential application of person verification, and proposes a general learning framework for the underlying fast similarity search problem. Finally, we make conclusions and explore future directions in Chapter 7.

CHAPTER 2

FROM PERSON IDENTIFICATION TO VERIFICATION

In this chapter, we discuss in detail the relationship between person verification and the more conventional problem, person identification. We shall see that the verification problem, to determine if two images in a pair are from the same person, is a more fundamental problem, while person identification can be tackled by solving multiple verification problems. Later in this chapter, we briefly review some of the representative works on person verification.

2.1 Limited-Sample Issue in Person Identification

The person identification problem has long been studied in the computer vision community. The general setup is shown in Figure 2.1. We have a set of people of interest, and each of them has a few images available. These images form a so-called *gallery set*. Now given a test image, also referred to as *the probe*, the goal is to decide which of the people in the gallery set this probe image comes from. In some cases, the system is also required to identify that the probe image is not from any of the known people, or in other words, is from an imposter.

From the machine learning perspective, the person identification problem is a standard classification problem. It can be solved by learning person dependent classification models (shown as M_1 – M_4 in Figure 2.1) from the gallery set. These models could be either generative, modeling the person-dependent image distributions, or discriminative classifiers such as k-Nearest Neighbors (k-NN) [4], AdaBoost [5], or Support Vector Machines (SVMs) [3].

In practice, however, we are often confronted with the so-called "limited-sample issue," as illustrated in Figure 2.2, which makes the traditional classification methods no long applicable. Specifically, people in the gallery set usually have very few images, or even a single image available. For example, it is common that only a single ID photo can be retrieved for a suspect at large. Or an access control

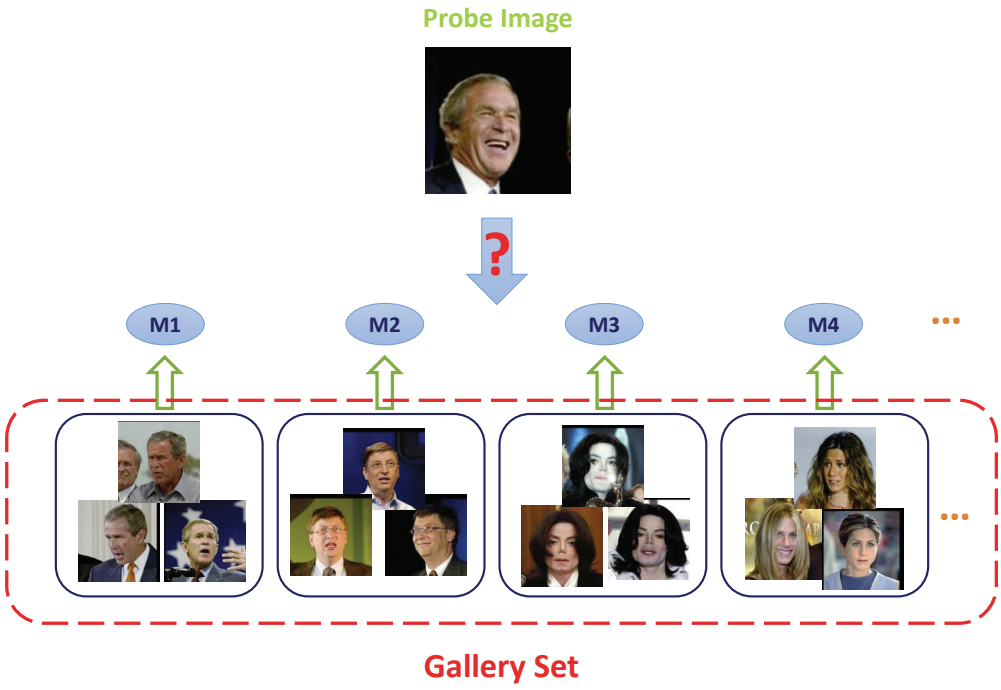


Figure 2.1: General setup for person identification.

system should be able to identify a person with granted access by just one or a few photos of him or her. In these cases, however, the above classification methods could hardly learn reliable person dependent models with a single or few images per person. Therefore, a traditional person identification system is likely to fail in many practical application scenarios.

2.2 Verification as Identification with a Single Example

The discussion above indicates that the person identification problem needs to be reconsidered from a machine learning perspective. Here we focus on the most challenging case – only one image per person in the gallery set – though generalization can be made in cases of more than one gallery images per person. Under this "one-gallery-image" condition, we could formally derive person verification as "identification with a single example."

We know that the classification problem can be understood as a hypothesis test.

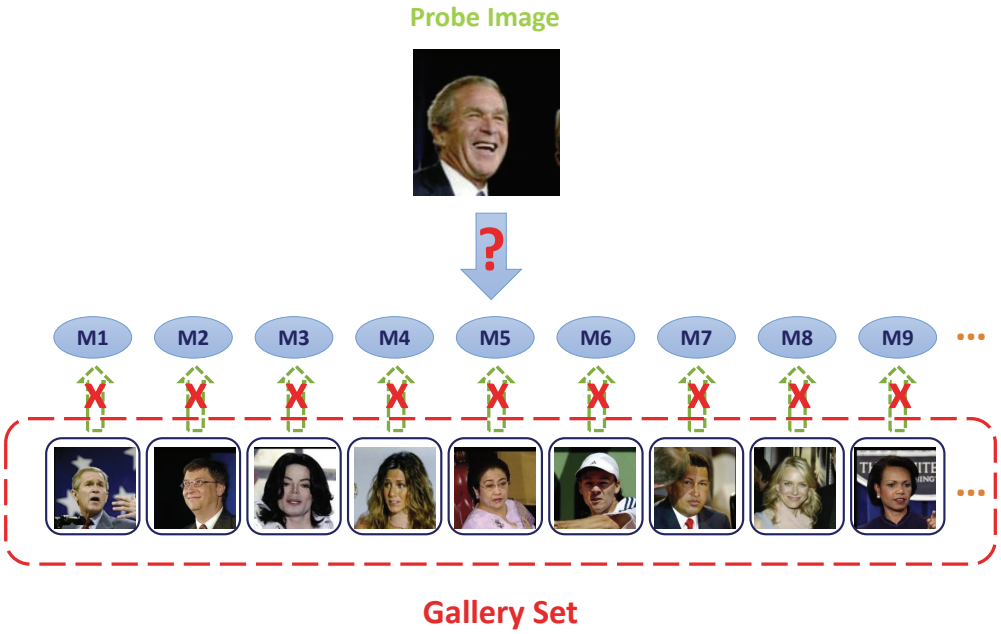


Figure 2.2: Illustration of the limited-sample issue in person identification, which often results in the failure of learning person dependent classification models.

Given a gallery set with s people, each with a single image, $\{x_1, \cdots, x_s\}$ (x_i denotes the image of the i-th person), and a probe image x_p, the $s+1$ hypotheses we consider are:

- $\mathcal{H}_i$: x_p is from the i-th person, for $1 \leq i \leq s$;
- $\mathcal{H}_0$: x_p is from an imposter.

According to the Bayes decision theory [4], the goal of person identification is to find the most likely hypothesis that has the largest data likelihood, namely,

$$\mathcal{H}^* = \arg\max_{0 \leq i \leq s} p(x_p, x_1, \cdots, x_s | \mathcal{H}_i)$$

To compute the data likelihood, one common assumption we can make is that the images are independently distributed conditioning on the class (identity). That means two images from different classes are independent random variables. So

under the i-th hypothesis, $1 \leq i \leq s$, the likelihood can be written as:

$$\begin{aligned} p(x_p, x_1, \cdots, x_s|\mathcal{H}_i) &= p(x_p, x_i|\mathcal{H}_i) \prod_{j=1, j\neq i}^{s} p(x_j) \\ &= \frac{p(x_p, x_i|\mathcal{H}_i)}{p(x_p)p(x_i)} p(x_p) \prod_{j=1}^{s} p(x_j) \\ &= \frac{p(x_p, x_i|\mathcal{H}_s)}{p(x_p, x_i|\mathcal{H}_d)} p(x_p) \prod_{j=1}^{s} p(x_j). \end{aligned}$$

Note that the third equality holds because under $\mathcal{H}_i$: (1) x_p and x_i is from the same class and hence $p(x_p, x_i|\mathcal{H}_i) = p(x_p, x_i|\mathcal{H}_s)$, and (2) $p(x_p, x_i|\mathcal{H}_d) = p(x_p)p(x_i)$ due to conditional independence. For hypothesis $\mathcal{H}_0$, since x_p is not from any of the gallery people, all gallery and probe images are independently distributed. Therefore,

$$p(x_p, x_1, \cdots, x_s|\mathcal{H}_0) = p(x_p) \prod_{j=1}^{s} p(x_j).$$

We immediately see that the most likely class can be determined by performing the verification tests for all possible pairs of a gallery image and the probe image, i.e.,

$$\mathcal{H}^* = \arg\max_{0 \leq i \leq s} \begin{cases} LLR(x_p, x_i), & \text{for } i \leq 1 \leq s; \\ 1, & \text{for } i = 0, \end{cases} \tag{2.1}$$

where $LLR(\cdot, \cdot)$ is given by (1.1).

In this way, we reduce the person identification problem to multiple verification problems, as shown in Figure 2.3. It can be understood in the following way. The identification problem is to find the most likely person from whom the probe image x_p comes. In the "one-gallery-image" case, each person has only one image in the gallery set, and thus we have $p(x_p|x_i, \mathcal{H}_i) \propto LLR(x_p, x_i) = \frac{p(x_p, x_i|\mathcal{H}_s)}{p(x_p, x_i|\mathcal{H}_d)}$ ($p(x_p)$ does not matter for classification).

The above observation indicates that person verification is a more fundamental problem we need to solve. And it can also be understood as "identification with a single example" because of $p(x_p|x_i, \mathcal{H}_i) \propto LLR(x_p, x_i)$.

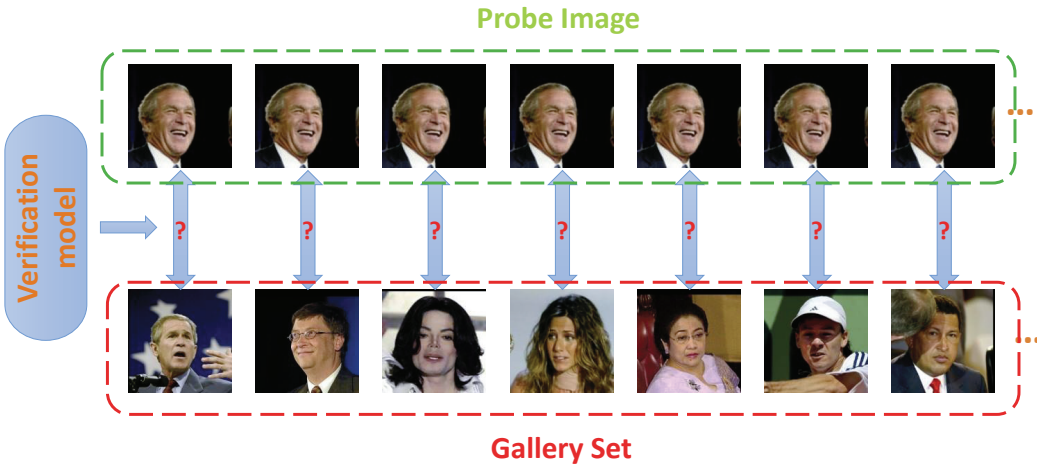


Figure 2.3: Person identification with one gallery image per person as multiple person verification problems.

2.3 Related Work on Person Verification

One of the early studies of the person verification problem is the Bayesian Face Recognition (BFR) work by Moghaddam et al. [2]. This work proposes a Bayesian approach similar to (1.1). However, one simplification the authors make is to model the difference vector $\Delta = x - y$, rather than having a joint model of x and y. They further assume that Δ follows zero-mean multivariate Gaussian distributions, with covariance Ω_I for *intrapersonal* variations (hypothesis $\mathcal{H}_s$), and covariance Ω_E for *extrapersonal* variations (hypothesis $\mathcal{H}_d$). The log-likelihood ratio is then given by:

$$LLR_{BFR}(\Delta) = \log \frac{p(\Delta|\Omega_I)}{p(\Delta|\Omega_E)} = \frac{1}{2}\Delta^T \left(\Omega_E^{-1} - \Omega_I^{-1}\right)\Delta + const \qquad (2.2)$$

One underlying assumption behind the BFR work is that the difference vector $\Delta = x - y$ captures most of the information to distinguish intrapersonal and extrapersonal variations, or in other words, image pairs from the same person and from different people. This assumption is nevertheless problematic and would lead to non-optimal decisions. We will discuss this in detail in Chapter 4.

More recently, many learning based approaches for person verification focus on learning a distance (similarity) measure between two images $d(x, y)$. Then one can decide whether x and y are from the same person based on some thresholding rule, i.e., $d(x, y) \leq d_0$.

One of the most straightforward distance measures is the Euclidean distance $\|x-y\|_2$. Without supervised learning, however, the Euclidean distance generally does not perform well in verification tasks. As a contrast, metric learning (ML) approaches [6] have been successfully applied to person verification [7, 8, 9], generating satisfactory performance. The key idea behind ML is to supervisedly learn a parametric distance metric, which in most cases takes the form of $(x-y)^T M(x-y)$ where M is a positive semi-definite (PSD) matrix.[1]

Although ML is very important for many supervised learning applications (e.g. classification) that often deal with complex and high-dimensional features, it has a few limitations particularly in the verification setting. The objective of many ML algorithms is to ensure that samples from the same class be closer to each other than those from different classes. In other words, it enforces a *relative* ranking constraint between intra-class and inter-class pairs (in terms of pairwise distances), and this is why ML is often tied with the nearest neighbor classifier for a classification task. However, for verification where many test samples might come from unseen classes, nearest neighbor classifiers are not applicable. Then ML only leads to an *absolute* decision rule with a constant threshold d_0:

$$f_{ML}(x,y) = d_0 - (x-y)^T M(x-y). \tag{2.3}$$

This intrinsic mismatch (classification vs. verification, relative ranking vs. absolute discrimination) leaves ML approaches not optimal for verification problems. We also note that decision function by ML in (2.3) depends on $x-y$ alone, and thus suffers from the same issue as BFR.

While ML gives a simple linear distance measure (on the original or kernel space), works exist that exploit complex, nonlinear, and less well-parameterized similarity measures between images. For instance, [10, 11] implement the visual similarity as an ensemble of ad-hoc, one-dimensional feature comparisons, via either randomized decision trees or AdaBoost. This type of method works well only with a collection of simple, low-dimensional, loosely coupled features, whereas state-of-the-art image/video features are usually complex and of very high dimensions [1].

Besides the above probabilistic models and distance based methods, another genre of approaches focus on extracting pairwise features (such that traditional

[1] A comprehensive review of ML methods can be found in the survey paper by Yang and Jin [6].

classifiers, e.g. SVM, can directly be applied). For instance, [12, 13, 14] construct pairwise features as the element-wise absolute difference and/or element-wise multiplication of two individual features. One shortcoming of these approaches is that the pairwise features are usually hand-crafted. And forcing the verification process into two disjoint steps, pairwise feature extraction and binary classification, often leaves the resulting verification model less optimized.

CHAPTER 3

MAXIMUM A POSTERIORI COVARIANCE VERIFICATION MODEL

In this chapter, we propose a parametric model for the verification problem under the Bayesian framework.

From the generative model point of view, it is common to assume that the images are independently distributed conditioning on the identity. Specifically, two images x_1 and x_2 are independent if they come from different classes (hypothesis $\mathcal{H}_d$), while under hypothesis $\mathcal{H}_s$, x_1 and x_2 are independent only if the underlying *identity model* is given.

Mathematically, the likelihood probabilities of the two hypotheses can be written as:

$$\begin{aligned} p(x_1, x_2|\mathcal{H}_s) &= \int p(x_1|m)p(x_2|m)p(m)dm, & (3.1)\\ p(x_1, x_2|\mathcal{H}_d) &= p(x_1)p(x_2) \\ &= \int p(x_1|m_1)p(m_1)dm_1 \int p(x_2|m_2)p(m_2)dm_2, & (3.2) \end{aligned}$$

where m, m_1, m_2 denote the underlying identity models. Substituting (3.1)-(3.2) into (1.1) leads to:

$$\begin{aligned} LLR(x_1, x_2) &= \log \frac{\int p(x_1|m)p(x_2|m)p(m)dm}{p(x_1)p(x_2)} \\ &= \log \frac{\int p(x_1|m)p(x_2|m)p(m)dm}{\int p(x_1|m_1)p(m_1)dm_1 \int p(x_2|m_2)p(m_2)dm_2}. \quad (3.3) \end{aligned}$$

3.1 Covariance Verification Model (CVM)

To compute the log-likelihood ratio in (3.3), the underlying models for the two distributions $p(m)$ and $p(x|m)$ need to be assumed. Inspired by [15], we start

with simple Gaussian models, namely,

$$m \sim \mathcal{N}(\mu, \Sigma_b) \tag{3.4}$$

$$x|m \sim \mathcal{N}(\mu + m, \Sigma_w) \tag{3.5}$$

Interestingly, (3.4)-(3.5) is equivalent to an additive, generative model, where each image is decomposed into two independent Gaussian distributed components: the *identity component* and the *noise component*. Denoting by x_{ij} the j-th image of the i-th person we have:

$$x_{ij} = m_i + e_{ij}, \tag{3.6}$$

where m_i is the identity component/model for the i-th person, and e_{ij} is the noise component, whose distribution, $\mathcal{N}(0, \Sigma_w)$, is assumed the same for all people. It is easy to show that, because of the independence between the identity component and the noise component, the additive model in (3.6) satisfies the conditional independence assumption.

Before going into details of model estimation and evaluation, we discuss a few interesting interpretations of the proposed model. First of all, $p(m) = \mathcal{N}(m|\mu, \Sigma_b)$ characterizes the extrapersonal variations, and $p(x|m) = \mathcal{N}(x|\mu+m, \Sigma_w)$ models the intrapersonal variations given the identity model m. Second, the distribution of images of each person is a Gaussian with tied covariance matrix Σ_w, and mean vector m uniquely defining the identity; at a higher level, the identity vectors of all people follow another Gaussian distribution with mean μ and covariance Σ_b. Finally, the marginal distribution of the images is also a Gaussian, given by:

$$p(x) = \int p(x|m)p(m)dm = \mathcal{N}(\mu, \Sigma_t), \tag{3.7}$$

where $\Sigma_t = \Sigma_b + \Sigma_w$. Because $\Sigma_t = \text{Cov}(x)$, $\Sigma_b = \text{Cov}(m)$, and $\Sigma_w = \text{Cov}(x - m(x))$ can be understood as the *total covariance*, *between-class covariance*, and *with-class covariance*, respectively, we term the proposed model in (3.4)-(3.5) or (3.6) as **Covariance Verification Model (CVM)**. The parameter set of CVM is denoted as $\theta = \{\mu, \Sigma_b, \Sigma_w\}$.

3.1.1 Scoring of CVM

In this section, we show that given learned model parameter θ, the log-likelihood can be evaluated in closed-form as follows:

$$LLR(x_1, x_2) = \frac{1}{2} {x'_1}^T A x'_1 + \frac{1}{2} {x'_2}^T A x'_2 + {x'_1}^T B x'_2 + const, \tag{3.8}$$

where $x'_1 = x_1 - \mu$, $x'_2 = x_2 - \mu$, and

$$A = \Sigma_t^{-1} - (\Sigma_t - \Sigma_b \Sigma_t^{-1} \Sigma_b)^{-1}, \tag{3.9}$$

$$B = \Sigma_t^{-1} \Sigma_b (\Sigma_t - \Sigma_b \Sigma_t^{-1} \Sigma_b)^{-1}. \tag{3.10}$$

Proof

Under hypothesis $\mathcal{H}_s$, x_1 and x_2 share the same identity component, i.e., $x_1 = m + e_1$ and $x_2 = m + e_2$. m, e_1, e_2 are independent Gaussian random variables, and thus

$$p(x_1, x_2|\mathcal{H}_s) = \mathcal{N}\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \middle| \begin{bmatrix} \mu \\ \mu \end{bmatrix}, \begin{bmatrix} \Sigma_t & \Sigma_b \\ \Sigma_b & \Sigma_t \end{bmatrix} \right). \tag{3.11}$$

Under hypothesis $\mathcal{H}_d$, x_1 and x_2 have their own identity components, i.e., $x_1 = m_1 + e_1$ and $x_2 = m_2 + e_2$. m_1, m_2, e_1 and e_2 are independent Gaussian random variables, and thus

$$p(x_1, x_2|\mathcal{H}_d) = \mathcal{N}\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \middle| \begin{bmatrix} \mu \\ \mu \end{bmatrix}, \begin{bmatrix} \Sigma_t & 0 \\ 0 & \Sigma_t \end{bmatrix} \right). \tag{3.12}$$

Therefore, the log-likelihood ratio in (3.3) can be rewritten as:

$$\begin{aligned} LLR(x_1, x_2) &= \log p(x_1, x_2|\mathcal{H}_s) - \log p(x_1, x_2|\mathcal{H}_d) \\ &= -\frac{1}{2} \begin{bmatrix} {x'_1}^T & {x'_2}^T \end{bmatrix} \begin{bmatrix} C^{-1} & \Sigma_t^{-1} \Sigma_b C^{-1} \\ C^{-1} \Sigma_b \Sigma_t^{-1} & C^{-1} \end{bmatrix} \begin{bmatrix} x'_1 \\ x'_2 \end{bmatrix} \\ &\quad + \frac{1}{2} \begin{bmatrix} {x'_1}^T & {x'_2}^T \end{bmatrix} \begin{bmatrix} \Sigma_t^{-1} & 0 \\ 0 & \Sigma_t^{-1} \end{bmatrix} \begin{bmatrix} x'_1 \\ x'_2 \end{bmatrix} + const \\ &= \frac{1}{2} {x'_1}^T A x'_1 + \frac{1}{2} {x'_2}^T A x'_2 + {x'_1}^T B x'_2 + const, \end{aligned} \tag{3.13}$$

where $C = \Sigma_t - \Sigma_b \Sigma_t^{-1} \Sigma_b$. Here we make use of the blockwise matrix inversion

identity [16]:

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} + A^{-1}B(D - CA^{-1}B)^{-1}CA^{-1} & -A^{-1}B(D - CA^{-1}B)^{-1} \\ (D - CA^{-1}B)^{-1}CA^{-1} & (D - CA^{-1}B)^{-1} \end{bmatrix}$$

and Woodbury matrix identity (matrix inverse lemma) [17]:

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}.$$

■

A few interesting properties of A and B are as follows:

(i) Both A and B are symmetric;

(ii) $\mathbf{rank}(A) = \mathbf{rank}(B) = \mathbf{rank}(\Sigma_b)$;

(iii) A is negative semi-definite (NSD) and B is positive semi-definite (PSD).

Proof

(i) Clearly A is symmetric. To show B is also symmetric, we first apply Woodbury matrix identity and obtain

$$\begin{aligned} \Sigma_b(I - \Sigma_t^{-1}\Sigma_b\Sigma_t^{-1}\Sigma_b)^{-1} &= \Sigma_b\left[I + \Sigma_t^{-1}\Sigma_b\Sigma_t^{-1}(I - \Sigma_b\Sigma_t^{-1}\Sigma_b\Sigma_t^{-1})^{-1}\Sigma_b\right] \\ &= \left[I + \Sigma_b\Sigma_t^{-1}\Sigma_b\Sigma_t^{-1}(I - \Sigma_b\Sigma_t^{-1}\Sigma_b\Sigma_t^{-1})^{-1}\right]\Sigma_b \\ &= (I - \Sigma_b\Sigma_t^{-1}\Sigma_b\Sigma_t^{-1})^{-1}\Sigma_b. \end{aligned}$$

Then

$$\begin{aligned} B = \Sigma_t^{-1}\Sigma_b C^{-1} &= \Sigma_t^{-1}\left[\Sigma_b(I - \Sigma_t^{-1}\Sigma_b\Sigma_t^{-1}\Sigma_b)^{-1}\right]\Sigma_t^{-1} \\ &= \Sigma_t^{-1}\left[(I - \Sigma_b\Sigma_t^{-1}\Sigma_b\Sigma_t^{-1})^{-1}\Sigma_b\right]\Sigma_t^{-1} \\ &= C^{-1}\Sigma_b\Sigma_t^{-1} = B^T. \end{aligned}$$

Hence B is symmetric.

(ii) It is trivial when Σ_b has full rank, in which case both A and B are full rank matrices. If Σ_b is rank deficient, for example, $\mathbf{rank}(\Sigma_b) = p < d$ where d is the original dimension, we show that $\mathbf{rank}(A) = \mathbf{rank}(B) = p$.

Let $\Sigma_b = FF^T, F \in \mathbb{R}^{d\times p}$ be a low-rank decomposition of Σ_b.[1] Again by Woodbury matrix identity we have

$$\begin{aligned} C^{-1} &= (\Sigma_t - FF^T\Sigma_t^{-1}FF^T)^{-1} \\ &= (\Sigma_t - FDF^T)^{-1} \\ &= \Sigma_t^{-1} + \Sigma_t^{-1}F(D^{-1} - D)^{-1}F^T\Sigma_t^{-1} \end{aligned}$$

where $D = F^T\Sigma_t^{-1}F$ is a rank-p matrix. Thus

$$A = \Sigma_t^{-1} - C^{-1} = -\Sigma_t^{-1}F(D^{-1} - D)^{-1}F^T\Sigma_t^{-1} \tag{3.14}$$

and

$$\begin{aligned} B = \Sigma_t^{-1}\Sigma_b C^{-1} &= \Sigma_t^{-1}FF^T\Sigma_t^{-1} + \Sigma_t^{-1}FF^T\Sigma_t^{-1}F(D^{-1} - D)^{-1}F^T\Sigma_t^{-1} \\ &= \Sigma_t^{-1}F(I + (D^{-2} - I)^{-1})F^T\Sigma_t^{-1} \\ &= \Sigma_t^{-1}F(I - D^2)^{-1}F^T\Sigma_t^{-1} \end{aligned} \tag{3.15}$$

are both of rank p.

(iii) Since $\begin{bmatrix} \Sigma_t & \Sigma_b \\ \Sigma_b & \Sigma_t \end{bmatrix}$ is the covariance matrix of the Gaussian distribution $p(x_1, x_2|\mathcal{H}_s)$, it must be positive definite, and so is its inverse. Thus we know

$$\begin{bmatrix} x_1^T & x_2^T \end{bmatrix} \begin{bmatrix} C^{-1} & \Sigma_t^{-1}\Sigma_b C^{-1} \\ C^{-1}\Sigma_b\Sigma_t^{-1} & C^{-1} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} > 0 \quad \forall x_1, x_2.$$

Simply setting $x_2 = 0$ leads to $x_1^T C^{-1} x_1 > 0$ for any x_1, which indicates that C is positive definite (and so is C^{-1}).

Therefore, A is negative semi-definite (NSD) because

$$\begin{aligned} A = \Sigma_t^{-1} - C^{-1} &= \Sigma_t^{-1} - \left[\Sigma_t^{-1} - \Sigma_t^{-1}\Sigma_b(I - \Sigma_b\Sigma_t^{-1}\Sigma_b)^{-1}\Sigma_b\Sigma_t^{-1}\right] \\ &= -\Sigma_t^{-1}\Sigma_b C^{-1}\Sigma_b\Sigma_t^{-1}. \end{aligned}$$

To show B is positive semi-definite, we need to rewrite (3.15) as

$$B = \Sigma_t^{-1}FD^{-\frac{1}{2}}(D^{-1} - D)^{-1}D^{-\frac{1}{2}}F^T\Sigma_t^{-1}.$$

[1] This can be obtained by eigen decomposition with all p non-zeros eigen values.

Since $D = F^T \Sigma_t^{-1} F$ is of full rank p, we immediately see that B is positive semi-definite (PSD) by comparing the above equation with A in (3.14).

■

3.1.2 Maximum a posteriori estimation of CVM using EM

In this section, we study algorithms to learn the parameters of the CVM. Suppose we are given a dataset of images:

$$\mathcal{X} = \left\{ x_{ij} \in \mathbb{R}^d : 1 \leq i \leq r, 1 \leq j \leq n_i \right\}, \tag{3.16}$$

where x_{ij} denotes the j-th image of the i-th person (with n_i images in total), and r is the number of people in this dataset. The total number of images is $n = \sum_{i=1}^{r} n_i$.

Given the probabilistic model in (3.4)-(3.5), it is intuitive to estimate the parameter $\theta = \{\mu, \Sigma_b, \Sigma_w\}$ under *maximum likelihood* (ML) criterion, i.e.,

$$\theta_{ML}^* = \arg\max_{\theta} \log p(\mathcal{X}|\theta). \tag{3.17}$$

However, substituting (3.4)-(3.5) into (3.17) under conditional independence assumption leads to no closed-form solution. Instead, we observe from (3.6) that the images of a particular person are associated with a same hidden variable. Therefore, we can regard the $p(\mathcal{X}|\theta)$ as the likelihood of the incomplete data (the hidden data denoted by $\mathcal{Z} = \{m_i : 1 \leq i \leq r\}$), and thus the well-known Expectation-Maximization (EM) algorithm [18] can be applied.

In principle, the EM algorithm is an iterative method for estimating the parameters of a statistical model by finding the maximum likelihood over a set of data samples with unobserved latent variables. The EM iterations alternate between an expectation (E) step and a maximization (M) step. In an E-step, it uses the current estimate for the parameters to compute the expectation of the log-likelihood, and in an M-step, it estimates the parameters by maximizing the expected log-likelihood found on the E-step. These parameter estimates are then used to determine the distribution of the latent variables in the next E-step. The convergence of this iterative algorithm is guaranteed.

Mathematically, EM algorithm iterates between the two following procedures:

- **Expectation step (E-step)**: Calculate the expected value of the log likelihood function, with respect to the conditional distribution of given under the current estimate of the parameters:

$$Q_{\text{ML}}(\theta|\theta^{(t)}) = \mathbb{E}_{\mathcal{Z}|\mathcal{X},\theta^{(t)}}\left[\log p(\mathcal{X},\mathcal{Z}|\theta)\right] \tag{3.18}$$

- **Maximization step (M-step)**: Find the parameter that maximizes this quantity:

$$\theta^{(t+1)} = \arg\max_{\theta} Q_{\text{ML}}(\theta|\theta^{(t)}) \tag{3.19}$$

The ML criterion used in conventional EM algorithm assumes that the size of the training data is large enough to provide robust parameter estimation. This is not always the case in practice, though. On a d-dimensional space, the number of parameters in CVM is $O(d^2)$. Common sense in machine learning requires the number of samples n be on the same order in order for a reliable parameter estimation, while in practice usually $n < d^2$. On the other hand, consider the parameter Σ_b that can be understood as the between-class covariance. From the knowledge in linear discriminant analysis (LDA) [4], between-class covariance is computed based on r class means only, and thus cannot be reliably estimated empirically due to the limited number of classes (people).

Motivated by [19], we adopt *maximum a posteriori* (MAP) estimation in the EM algorithm. The idea of MAP estimation is to leverage on some prior knowledge of parameters rather than fully relying on the training data. In this method, the parameters to be estimated are regarded as random variables whose prior probability is assumed. Instead of maximizing the log-likelihood in (3.17), MAP estimator tries to maximize the posterior probability of the parameters given the training data, i.e.,

$$\begin{aligned} \theta^{*}_{\text{MAP}} &= \arg\max_{\theta} \log p(\theta|\mathcal{X},\Phi) \\ &= \arg\max_{\theta} \log p(\mathcal{X}|\theta)p(\theta|\Phi), \end{aligned} \tag{3.20}$$

where $p(\theta|\Phi)$ is the prior distribution of θ, parameterized by Φ. The according change to the original EM algorithm is to augment the $Q(\cdot)$ function in (3.18)-

(3.19) with an additional term $\log p(\theta|\Phi)$, i.e.,

$$\begin{aligned} Q_{\text{MAP}}(\theta|\theta^{(t)}) &= \mathbb{E}_{\mathcal{Z}|\mathcal{X},\theta^{(t)}}\left[\log p(\mathcal{X},\mathcal{Z},\theta|\Phi)\right] \\ &= Q_{\text{ML}}(\theta|\theta^{(t)}) + \log p(\theta|\Phi). \end{aligned} \tag{3.21}$$

The MAP estimation provides a framework for incorporating prior information in the training process. This is particularly useful in dealing with problems posed by sparse training data when the ML approach fails to give accurate estimates of the parameters. From another point of view, MAP estimation can be regarded as a way of parameter smoothing/interpolation. It is well-known that the MAP estimates are asymptotically convergent to the ML ones [4]. As the amount of training data increases from zero to infinity, the MAP approach will generate parameter estimates that smoothly vary from the priors to the ML estimates.

What remains now is the choice of prior distribution family and the specification of the parameters of prior densities. It is common to assume the conjugate prior such that the prior and posterior distribution are of the same distribution family [4]. The conjugate prior for multivariate Gaussian with known mean (3.5) is an *Inverse-Wishart* distribution:

$$\begin{aligned} p(\Sigma_w|\Psi_w,\nu_w) &= \mathcal{W}^{-1}(\Sigma_w|\Psi_w,\nu_w) \\ &\propto |\Sigma_w|^{-\frac{\nu_w+d+1}{2}} \exp\left\{-\frac{1}{2}\mathbf{tr}(\Psi_w\Sigma_w^{-1})\right\}, \end{aligned} \tag{3.22}$$

and for multivariate Gaussian with unknown mean and covariance (3.4) is a *Normal-Inverse-Wishart* distribution:

$$\begin{aligned} p(\mu,\Sigma_b|\mu_0,\beta,\Psi_b,\nu_b) &= \mathcal{N}(\mu|\mu_0,\beta\Sigma_b)\cdot\mathcal{W}^{-1}(\Sigma_b|\Psi_b,\nu_b) \\ &\propto |\Sigma_b|^{-\frac{\nu_b+d+2}{2}} \exp\left\{-\frac{1}{2\beta}(\mu-\mu_0)^T\Sigma_b^{-1}(\mu-\mu_0)\right\} \\ &\quad\cdot\exp\left\{-\frac{1}{2}\mathbf{tr}(\Psi_b\Sigma_b^{-1})\right\}, \end{aligned} \tag{3.23}$$

where $\mu_0 \in \mathbb{R}^d$, $\beta > 0$, Ψ_b, Ψ_w are $d \times d$ PSD matrices, and $\nu_b, \nu_w > d-1$ are the real-valued degree of freedom of the two Wishart distribution, respectively. We leave the choice of these parameter in later discussion.

To distinguish with CVM by conventional maximum likelihood EM algorithm, we name the model estimated under MAP criterion as **MAP-CVM**. Without proof we show that the EM steps for MAP-CVM can be computed as follows.

- E-step:

$$\mathbb{E}[m_i|\mathcal{X}] = \mu + (\Sigma_b^{-1} + n_i\Sigma_w^{-1})^{-1}\Sigma_w^{-1}\sum_{j=1}^{n_i}(x_{ij} - \mu), \quad (3.24)$$

$$\text{Cov}[m_i|\mathcal{X}] = (\Sigma_b^{-1} + n_i\Sigma_w^{-1})^{-1}. \quad (3.25)$$

We denote $u_i = \mathbb{E}[m_i|\mathcal{X}]$ and $V_i = \mathbb{E}[m_i m_i^T|\mathcal{X}] = \text{Cov}[m_i|\mathcal{X}] + u_i u_i^T$.

- M-step:

$$\mu^* = \alpha_\mu \bar{u} + (1 - \alpha_\mu)\mu_0, \quad (3.26)$$

$$\Sigma_b^* = \alpha_b \left[\bar{V} - 2\mu^*\bar{u}^T + \mu^*\mu^{*T}\right] + (1 - \alpha_b)\Psi_b, \quad (3.27)$$

$$\Sigma_w^* = \alpha_w \frac{1}{n}\left[\sum_{ij} x_{ij}x_{ij}^T - 2\sum_{ij} x_{ij}u_i^T + \sum_i n_i V_i\right] + (1 - \alpha_w)\Psi_w, \quad (3.28)$$

where

$$\bar{u} = \frac{1}{r}\sum_i u_i, \quad \bar{V} = \frac{1}{r}\sum_i V_i, \quad (3.29)$$

$$\alpha_\mu = \frac{r}{r + \gamma_\mu}, \quad \alpha_b = \frac{r}{r + \gamma_b}, \quad \alpha_w = \frac{n}{n + \gamma_w}, \quad (3.30)$$

$$\gamma_\mu = \frac{1}{\beta}, \quad \gamma_b = \nu_b + d + 2, \quad \gamma_w = \nu_w + d + 1. \quad (3.31)$$

We should note that the parameters reestimated in the M-step are interpolations between the ML estimates and the priors given by μ_0, Ψ_b, and Ψ_w. Moreover, the weights on the ML estimates, given by (3.30), increase as the number of people r and the number of images n increase. In an extreme case where both r and n are indefinitely large, the prior terms vanish and the final estimates become the ML ones. On the other hand, when only limited data is available, the estimation procedure tends to put more "trust" in the prior information.

Also note that the six prior parameters $\{\mu_0, \Psi_b, \Psi_w\}$ and $\{\gamma_\mu, \gamma_b, \gamma_w\}$ need to be specified in the learning phase. While the MAP framework provides no specific ways to calculate these parameters, optimal values have to be determined empirically. The former set of parameters provides prior knowledge about the CVM parameters to be estimated, and the latter set controls the smoothing/interpolation coefficients. In practice, $\{\mu_0, \Psi_b, \Psi_w\}$ can be inferred from some other datasets,

or we can simply set $\mu_0 = 0$ and $\Psi_b = \Psi_w = I$. The controlling parameters $\{\gamma_\mu, \gamma_b, \gamma_w\}$, on the other hand, can be determined via cross-validation.

3.1.3 Closed-form solutions to CVM

In this section we provide an alternative method to estimate the parameters of CVM. This method leads to closed-form solutions, which are much more efficient than the above EM algorithm. Moreover, the derived closed-form solutions well approximate the true ML or MAP estimation, and become exact under many common conditions. In practice, we find that the close-form solutions always yield comparable performance to the EM algorithm.

Given the fact that the parameters in CVM, μ, Σ_b, and Σ_w, implicitly characterize the data *mean*, *between-class covariance*, and *within-class covariance* respectively, one may naturally wonder if it is possible to compute them analytically. One choice is to use the empirical estimation, i.e.,

$$\tilde{\mu} = \frac{1}{n}\sum_{i=1}^{r}\sum_{j=1}^{n_i} x_{ij}, \tag{3.32}$$

$$\tilde{\Sigma}_w = \frac{1}{n}\sum_{i=1}^{r}\sum_{j=1}^{n_i}(x_{ij}-\tilde{\mu}_i)(x_{ij}-\tilde{\mu}_i)^T, \tag{3.33}$$

$$\tilde{\Sigma}_b = \frac{1}{n}\sum_{i=1}^{r} n_i(\tilde{\mu}_i-\tilde{\mu})(\tilde{\mu}_i-\tilde{\mu})^T, \tag{3.34}$$

where $\tilde{\mu}_i = \frac{1}{n_i}\sum_{j=1}^{n_i} x_{ij}$ is the empirical mean of images from the i-th person. Note that this is the same empirical estimation used in LDA [4]. As we shall see next, it is in fact an approximate solution of CVM under the maximum likelihood (ML) criterion. We also derive similar results for maximum a posteriori (MAP) estimation.

Recall that in order to obtain the ML estimates, one needs to optimize the objective function in (3.17). Considering the conditional independence of the images of the same person on their shared identity component (3.5), we can break down

the log-likelihood as follows:

$$\begin{aligned} L_{\text{ML}} &= \log p(\mathcal{X}|\theta) = \sum_{i=1}^{r} \log p(\mathcal{X}_i|\theta) \\ &= \sum_{i=1}^{r} \log \int \prod_{j=1}^{n_i} \left[\int p(x_{ij}|m_i, \Sigma_w) dx_{ij} \right] p(m_i|\mu, \Sigma_b) dm, \end{aligned} \quad (3.35)$$

where $\mathcal{X}_i = \{x_{i1}, \cdots, x_{in_i}\}$ denotes the set of images belonging to the i-th person.

At the first glance, directly optimizing (3.35) seems to be intractable. However, we note that both $p(m_i|\mu, \Sigma_b)$ (3.4) and $p(x_{ij}|m_i, \Sigma_w)$ (3.5) are Gaussian PDFs, which indicates that a closed form can be obtained for the above integrations. In fact, we could approach this problem in a more straightforward way.

Consider a new random variable as the concatenation of all n_i images from the i-th person, namely,

$$\boldsymbol{X}_i = \begin{bmatrix} x_{i1} \\ \vdots \\ x_{in_i} \end{bmatrix} = \begin{bmatrix} m_i + e_{i1} \\ \vdots \\ m_i + e_{in_i} \end{bmatrix} \in \mathbb{R}^{n_i d}.$$

Since m_i and e_{ij}'s are independent Gaussian random variables, $\mathbf{X}_i$ is also Gaussianly distributed: $\boldsymbol{X}_i \sim \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, with mean and covariance as follows:

$$\begin{aligned} \boldsymbol{\mu}_i &= \begin{bmatrix} \mu \\ \vdots \\ \mu \end{bmatrix} = \boldsymbol{I}_{n_i} \mu, \\ \boldsymbol{\Sigma}_i &= \begin{bmatrix} \Sigma_b + \Sigma_w & \Sigma_b & \cdots & \Sigma_b \\ \Sigma_b & \Sigma_b + \Sigma_w & \cdots & \Sigma_b \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_b & \Sigma_b & \cdots & \Sigma_b + \Sigma_w \end{bmatrix} \\ &= \mathbf{Diag}_{n_i}(\Sigma_w) + \boldsymbol{I}_{n_i} \Sigma_b \boldsymbol{I}_{n_i}^T, \end{aligned}$$

where $\boldsymbol{I}_n = \underbrace{[I \ \cdots \ I]}_{n}^T$ and $\mathbf{Diag}_n(A) = \underbrace{\begin{bmatrix} A & 0 & \cdots & 0 \\ 0 & A & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A \end{bmatrix}}_{n}$.

Then the log-likelihood in (3.35) can be simplified as:

$$\begin{aligned} L_{\text{ML}} &= \sum_{i=1}^{r} \mathcal{N}(\boldsymbol{X}_i|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \\ &= -\frac{1}{2}\sum_{i=1}^{r} \log|\boldsymbol{\Sigma}_i| - \frac{1}{2}\sum_{i=1}^{r}(\boldsymbol{X}_i - \boldsymbol{\mu}_i)^T\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{X}_i - \boldsymbol{\mu}_i) + const. \end{aligned} \tag{3.36}$$

The first term on the righthand side can be computed as follows:

$$\begin{aligned} \sum_{i=1}^{r} \log|\boldsymbol{\Sigma}_i| &= \sum_{i=1}^{r} \log\Big|\ \mathbf{Diag}_{n_i}(\Sigma_w) + \boldsymbol{I}_{n_i}\Sigma_b\boldsymbol{I}_{n_i}^T\ \Big| \\ &= \sum_{i=1}^{r} \log\Big|\ \mathbf{Diag}_{n_i}(\Sigma_w)\ \Big|\Big|\ I + \Sigma_b\boldsymbol{I}_{n_i}^T\mathbf{Diag}_{n_i}(\Sigma_w)^{-1}\boldsymbol{I}_{n_i}\ \Big| \\ &= \sum_{i=1}^{r} \log|\Sigma_w|^{n_i}|I + n_i\Sigma_b\Sigma_w^{-1}| \\ &= \sum_{i=1}^{r} \big[(n_i - 1)\log|\Sigma_w| + \log|\Sigma_w + n_i\Sigma_b|\big] \\ &= (n - r)\log|\Sigma_w| + \sum_{i=1}^{r}\log|\Sigma_w + n_i\Sigma_b|. \end{aligned} \tag{3.37}$$

To compute the second-term, we first apply Woodbury matrix identity to the inversion of $\boldsymbol{\Sigma}_i$:

$$\begin{aligned} \boldsymbol{\Sigma}_i^{-1} &= \mathbf{Diag}_{n_i}(\Sigma_w)^{-1} - \mathbf{Diag}_{n_i}(\Sigma_w)^{-1}\boldsymbol{I}_{n_i}(\Sigma_b^{-1} + n_i\Sigma_w^{-1})^{-1}\boldsymbol{I}_{n_i}^T\mathbf{Diag}_{n_i}(\Sigma_w)^{-1} \\ &= \mathbf{Diag}_{n_i}(\Sigma_w)^{-1} - \boldsymbol{I}_{n_i}\Sigma_w^{-1}(\Sigma_b^{-1} + n_i\Sigma_w^{-1})^{-1}\Sigma_w^{-1}\boldsymbol{I}_{n_i}^T. \end{aligned}$$

Thus,

$$
\begin{aligned}
& (\boldsymbol{X}_i - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\boldsymbol{X}_i - \boldsymbol{\mu}_i) \\
= & \sum_j^{n_i} (x_{ij} - \mu)^T \Sigma_w^{-1} (x_{ij} - \mu) - n_i^2 (\tilde{\mu}_i - \mu)^T \Sigma_w^{-1} (\Sigma_b^{-1} + n_i \Sigma_w^{-1})^{-1} \Sigma_w^{-1} (\tilde{\mu}_i - \mu) \\
= & \mathbf{tr}\Big[\Sigma_w^{-1} \sum_j^{n_i} (x_{ij} - \mu)(x_{ij} - \mu)^T\Big] \\
& -\mathbf{tr}\Big[n_i^2 \Sigma_w^{-1} (\Sigma_b^{-1} + n_i \Sigma_w^{-1})^{-1} \Sigma_w^{-1} (\tilde{\mu}_i - \mu)(\tilde{\mu}_i - \mu)^T\Big] \\
= & \mathbf{tr}\Big[\Sigma_w^{-1} (\sum_j^{n_i} (x_{ij} - \mu)(x_{ij} - \mu)^T - n_i (\tilde{\mu}_i - \mu)(\tilde{\mu}_i - \mu)^T)\Big] \\
& + n_i \mathbf{tr}\Big[(\Sigma_w^{-1} - n_i \Sigma_w^{-1} (\Sigma_b^{-1} + n_i \Sigma_w^{-1})^{-1} \Sigma_w^{-1}) (\tilde{\mu}_i - \mu)(\tilde{\mu}_i - \mu)^T\Big] \\
= & \mathbf{tr}\Big[\Sigma_w^{-1} \sum_j^{n_i} (x_{ij} - \tilde{\mu}_i)(x_{ij} - \tilde{\mu}_i)^T\Big] + n_i \mathbf{tr}\Big[(\Sigma_w + n_i \Sigma_b)^{-1} (\tilde{\mu}_i - \mu)(\tilde{\mu}_i - \mu)^T\Big].
\end{aligned}
$$

Then the second term on the righthand side of (3.36) becomes:

$$
\begin{aligned}
& \sum_{i=1}^{r} (\boldsymbol{X}_i - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\boldsymbol{X}_i - \boldsymbol{\mu}_i) \\
= & \; n \mathbf{tr}(\Sigma_w^{-1} \tilde{\Sigma}_w) + \mathbf{tr}\Big[\sum_{i=1}^{r} n_i (\Sigma_w + n_i \Sigma_b)^{-1} (\tilde{\mu}_i - \mu)(\tilde{\mu}_i - \mu)^T\Big]. \quad (3.38)
\end{aligned}
$$

Now we can rewrite the log-likelihood by substituting (3.37) and (3.38) into (3.36):

$$
\begin{aligned}
L_{\text{ML}} = & -\frac{1}{2}(n - r) \log |\Sigma_w| - \frac{1}{2} \sum_{i=1}^{r} \log |\Sigma_w + n_i \Sigma_b| - \frac{n}{2} \mathbf{tr}\Big[\Sigma_w^{-1} \tilde{\Sigma}_w\Big] \\
& -\frac{1}{2} \mathbf{tr}\Big[\sum_{i=1}^{r} n_i (\Sigma_w + n_i \Sigma_b)^{-1} (\tilde{\mu}_i - \mu)(\tilde{\mu}_i - \mu)^T\Big] + const. \quad (3.39)
\end{aligned}
$$

To find the optimal parameter set $\theta = \{\mu, \Sigma_b, \Sigma_w\}$ that maximizes the data log-likelihood, we need to find the stationary point of L_{ML}. So we take the derivatives of L_{ML} with respect to μ, Σ_b, and Σ_w respectively, and equate them to zero. This

leads to the following stationary point equations:

$$\frac{\partial L_{\text{ML}}}{\partial \mu} = \sum_{i=1}^{r} n_i(\Sigma_w + n_i\Sigma_b)^{-1}(\tilde{\mu}_i - \mu) = 0, \tag{3.40}$$

$$\begin{aligned}\frac{\partial L_{\text{ML}}}{\partial \Sigma_w} &= \frac{n}{2}\Sigma_w^{-1}\tilde{\Sigma}_w\Sigma_w^{-1} - \frac{1}{2}(n-r)log|\Sigma_w| \\ &+\frac{1}{2}\sum_{i=1}^{r} n_i(\Sigma_w + n_i\Sigma_b)^{-1}(\tilde{\mu}_i - \mu)(\tilde{\mu}_i - \mu)^T(\Sigma_w + n_i\Sigma_b)^{-1} \\ &-\frac{1}{2}\sum_{i=1}^{r}(\Sigma_w + n_i\Sigma_b)^{-1} = 0,\end{aligned} \tag{3.41}$$

$$\begin{aligned}\frac{\partial L_{\text{ML}}}{\partial \Sigma_b} &= \frac{1}{2}\sum_{i=1}^{r} n_i^2(\Sigma_w + n_i\Sigma_b)^{-1}(\tilde{\mu}_i - \mu)(\tilde{\mu}_i - \mu)^T(\Sigma_w + n_i\Sigma_b)^{-1} \\ &-\frac{1}{2}\sum_{i=1}^{r} n_i(\Sigma_w + n_i\Sigma_b)^{-1} = 0.\end{aligned} \tag{3.42}$$

While directly solving (3.40)-(3.42) is non-trivial, we can show that the empirical solution, given by (3.32)-(3.34), is a good approximation to the optimal θ. More formally we have:

Proposition 3.1 *The empirical estimates of μ, Σ_w and Σ_b, given by* (3.32)*,* (3.33) *and* (3.34) *respectively, asymptotically converge to the ML estimates as $n_i \to \infty$, $\forall i$.*

Proof As $n_i \to \infty$, $(\Sigma_w + n_i\Sigma_b)^{-1} \to 0$ and $n_i(\Sigma_w + n_i\Sigma_b)^{-1} \to \Sigma_b^{-1}$. Substituting them into (3.40) leads to $\mu^* = \sum_{i=1}^{r} \tilde{\mu}_i \to \tilde{\mu}$. Similarly, based on (3.41) and (3.42) we can arrive at $\Sigma_w^* = \tilde{\Sigma}_w$ and $\Sigma_b^* = \frac{1}{r}\sum_{i=1}^{r}(\tilde{\mu}_i - \tilde{\mu})(\tilde{\mu}_i - \tilde{\mu})^T \to \tilde{\Sigma}_b$, respectively. ■

In fact, the asymptotical convergence can be understood intuitively. Suppose there are an infinite number of samples for each person ($n_i \to \infty$, $\forall i$). Based on the law of large numbers (LLN) in probability theory, it is then safe to claim that the empirical mean of the i-th person, $\tilde{\mu}_i = \frac{1}{n_i}\sum_{i=1}^{r} x_{ij}$, gives the true class mean, or the identity component m_i, i.e., $\tilde{\mu}_i \to m_i$ as $n_i \to \infty$. In this way the hidden variables m_i's are exactly inferred without resorting to the EM algorithm. Therefore we can separately estimate the distribution of the identity component, parameterized by μ and Σ_b, and that of the noise term, parameterized by Σ_w. And the solution is the same as the empirical estimation in (3.32)-(3.34).

Balanced-classes case

In addition to the empirical estimation, we further provide a more "accurate" approximation to the true ML estimation. This approximate solution becomes exact not only asymptotically, but also in cases where the classes are well-balanced, i.e., consisting of a equal number of samples. In the latter case, we obtain the following closed-form solution to the stationary point equations in (3.40)-(3.41):

$$\begin{aligned} \mu^* &= \tilde{\mu}, \\ \Sigma_w^* &= \frac{n}{n-r}\tilde{\Sigma}_w, \\ \Sigma_b^* &= \tilde{\Sigma}_b - \frac{r}{n-r}\tilde{\Sigma}_w. \end{aligned}$$

Note that the above estimates are closely related to the empirical ones. First, both lead to the same estimate for the total covariance, namely, $\Sigma_t^* = \Sigma_w^* + \Sigma_b^* = \tilde{\Sigma}_b + \tilde{\Sigma}_w = \tilde{\Sigma}_t$. Second, as the number of samples n grows, the above estimates also converge to the empirical ones.

In practice, we observe minimal difference between the above estimates, and the empirical ones in (3.32)-(3.34), and those by EM algorithm. This is because, for one thing, people tend to use balanced training sets to avoid being biased toward the classes with dominant number of samples, and for the second, the number of samples is usually much larger than the number of people in the dataset ($n \gg r$). (It is a lot easier to collect more images for each person rather than collecting images from more people.)

Closed forms for MAP-CVM

The discussion above focuses on finding closed-form solutions to CVM under the maximum likelihood criterion. However, as we discuss in Section 3.1.2, MAP estimation, as a way of incorporating prior information in the training process, usually provides more reliable parameter estimates than ML, especially when training data is limited. So it would be of great interest to derive closed-form solutions for MAP-CVM as well.

In MAP estimation we seek the optimal parameter set $\theta = \{\mu, \Sigma_b, \Sigma_w\}$ that maximizes the posterior probability in (3.20) instead of the likelihood in (3.17).

Mathematically, the objective for MAP estimation is:

$$\begin{aligned} L_{\text{MAP}} &= \log p(\mathcal{X}|\theta)p(\theta|\Phi) \\ &= L_{\text{ML}} + \log p(\Sigma_w|\Psi_w, \nu_w) + \log p(\mu, \Sigma_b|\mu_0, \beta, \Psi_b, \nu_b) \\ &= L_{\text{ML}} - \frac{\gamma_w}{2}\log|\Sigma_w| - \frac{1}{2}\mathbf{tr}(\Sigma_w^{-1}\Psi_w) - \frac{\gamma_b}{2}\log|\Sigma_b| \\ &\quad -\frac{\gamma_\mu}{2}\mathbf{tr}\left[\Sigma_b^{-1}(\mu-\mu_0)(\mu-\mu_0)^T\right] - \frac{1}{2}\mathbf{tr}(\Sigma_b^{-1}\Psi_b), \end{aligned} \tag{3.43}$$

where $p(\theta|\Phi)$ is the prior distribution of θ specified in (3.22) and (3.23), and γ_μ, γ_b and γ_w are given by (3.31). Thus its stationary point equations are given by:

$$\begin{aligned} \frac{\partial L_{\text{MAP}}}{\partial \mu} &= \frac{\partial L_{\text{ML}}}{\partial \mu} + \gamma_\mu \Sigma_b^{-1}(\mu_0 - \mu) = 0, \\ \frac{\partial L_{\text{MAP}}}{\partial \Sigma_w} &= \frac{\partial L_{\text{ML}}}{\partial \Sigma_w} - \frac{\gamma_w}{2}\Sigma_w^{-1} + \frac{1}{2}\Sigma_w^{-1}\Psi_w\Sigma_w^{-1} = 0, \\ \frac{\partial L_{\text{MAP}}}{\partial \Sigma_b} &= \frac{\partial L_{\text{ML}}}{\partial \Sigma_b} - \frac{\gamma_b}{2}\Sigma_b^{-1} \\ &\quad + \frac{1}{2}\Sigma_b^{-1}\left[\gamma_\mu(\mu-\mu_0)(\mu-\mu_0)^T + \Psi_b\right]\Sigma_b^{-1} = 0. \end{aligned}$$

Similarly, we could not directly find an analytic solution to the above equations. Instead, we seek an approximation by letting $m_i \approx \tilde{\mu}_i$, which is true when n_i is large. Then the following analytic solution can be obtained for MAP-CVM:

$$\begin{aligned} \hat{\mu} &= \tilde{\mu}, \\ \hat{\Sigma}_w &= \alpha_w\tilde{\Sigma}_w + (1-\alpha_w)\Psi_w, \\ \hat{\Sigma}_b &= \alpha_b\tilde{\Sigma}_b + (1-\alpha_b)\Psi_b, \end{aligned}$$

where α_w and α_b are given by (3.30). Note that we simply use the empirical estimate for μ and disregard its prior. This is because we find in practice that the global mean can always be reliably estimated, given $n = \sum_i n_i$ while n_i is large. The above MAP estimation is naturally a smoothed version between the empirical solution, obtained under ML criterion, and the prior. As the size of the training set grows, α_w and α_b both go to 1, and thus the above degenerates to the empirical estimation. On the other hand, when only limited data is available, the estimation procedure tends to put more "trust" in the prior information.

Again, the prior parameters $\{\Psi_b, \Psi_w\}$ and $\{\gamma_b, \gamma_w\}$ need to specified in the learning phase. The former set of parameters provides prior knowledge about

Table 3.1: Statistics of the selected subset from Multi-PIE.

# images/person	# people	# images
< 100	7	451
$[100, 200)$	87	12575
$[200, 300)$	53	13626
$[300, 400)$	64	21859
≥ 400	126	56676
total	337	105,187

the CVM parameters to be estimated, and the latter set controls the smoothing/interpolation coefficients. In practice, we find that simply setting $\Psi_b = \Psi_w = I$ works pretty well. The controlling parameters $\{\gamma_b, \gamma_w\}$, on the other hand, can be determined via cross-validation.

3.2 Experiments

In this section, we evaluate the proposed approach on the CMU Multi-PIE dataset [20] and the Labeled Face in the Wild (LFW) dataset [21].

3.2.1 Multi-PIE

The CMU Multi-PIE database [20] is one of the largest face databases involving both huge extrapersonal and intrapersonal variations. It consists of 337 subjects, with more than 750,000 images captured under 15 viewpoints and 19 illumination conditions while displaying a range of facial expressions. The images are taken in up to four sessions over the span of five months.

We select a subset of 105,187 images for evaluation. These images are from 5 near-frontal views, from -30° to 30° with a 15° interval (numbered as 13_0, 14_0, 05_1, 05_0, and 04_1), and 13 ordinary (no side light) lighting conditions (numbered as 04, 05, 06, 07, 08, 09, 10, 14, 15, 16, 17, 18, and 19). Figure 3.1 shows the same subject under the selected poses and illumination conditions (all with neutral expression). We choose only neutral and smile expressions from all four sessions. Since not all subjects appear in all four sessions, and also due to the failure of the face detector, the number of face images per person ranges from 63 to 455. Some detailed statistics are listed in Table 3.1.

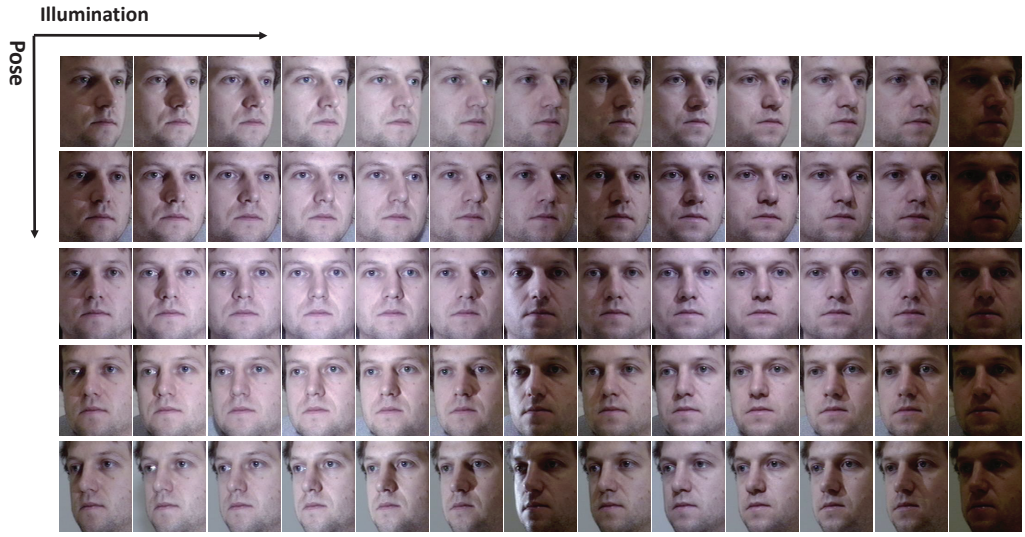


Figure 3.1: Example images of Multi-PIE dataset, showing the same subject under the 5 selected poses and 13 selected illumination conditions (all with neutral expression).

In the experiments, we partition the dataset into two disjoint parts in terms of identities. First, to ensure enough intrapersonal variations, we keep only 243 people with more than 200 images each (92161 images in total). We then randomly select a fixed set of 30 people as the testing set. In the remaining data, we further sample $r = \{25, 50, 100, 150, 200\}$ people, each with r images, to form the training set.

The evaluation protocol is as follows. The parameters of CVM are learned on the training set. Since minimal difference is observed between the EM algorithm and the closed-form solution (for both ML-CVM and MAP-CVM), we use the closed-form estimation in all experiments due to its efficiency. In testing, we draw positive (same-person) and negative (different-person) image pairs from the test set, and compute for each pair the log-likelihood ratio in (3.8). We report the accuracy (recall rate) at 1% false alarm rate on the ROC curve. All results are averaged over 10 random runs.

We employ the Hierarchical Gaussianization (HG) [1] as image features. The HG representation has generated many state-of-the-art results in a number of visual recognition tasks, including object categorization [1], person identification [1], age/gender estimation [22, 23], video event recognition [24], etc. The detailed feature extraction procedure is as follows. First each face image is resize

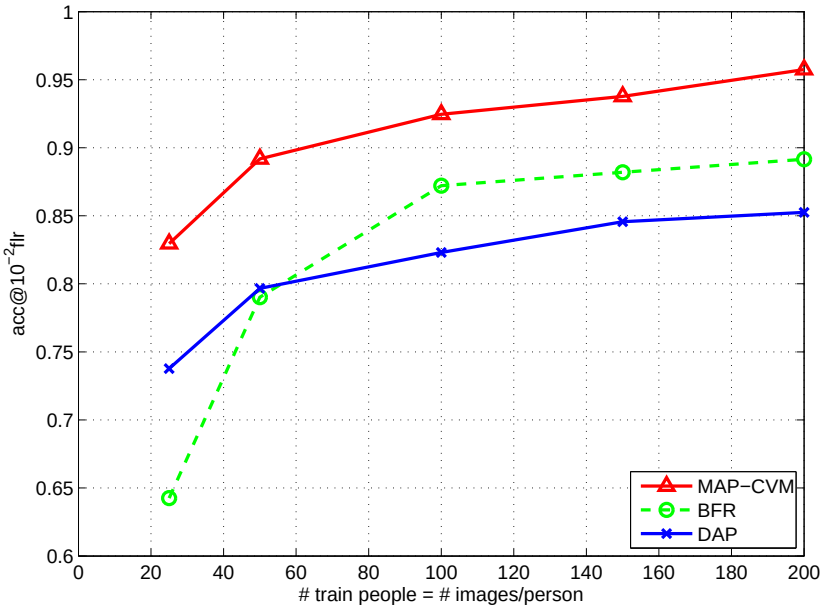


Figure 3.2: Comparison of MAP-CVM with previous methods on Multi-PIE dataset: DAP [1], and BFR [2].

to a resolution of 160×160 pixels. Then we extract SIFT [25] descriptors on a dense grid of overlapping patches, with patch size being 16, 24, and 32 and grid spacing being 4 pixels; the 128-dim SIFT descriptors are subsequently projected to 50 dimensions with PCA. Based on the local SIFT descriptors, we obtain the 512-mixture HG features, whose dimensionality is further reduced from 26624 to 200 by PCA.

Figure 3.2 compares the proposed MAP-CVM with Bayesian Face Recognition (BFR) [2], and Discriminant Attribute Projection (DAP) in the original HG paper [1]. It can be observed that our proposed method consistently outperforms previous works while the number of people in the training set varies from 25 to 200 (the number of images varies from 625 to 40000). Note the baseline accuracy by Euclidean distance is only 27.87%.

We are also interested in how much the MAP training improves over traditional ML. So we compare MAP-CVM with ML-CVM by setting $\gamma_w = \gamma_b = 0$ in (3.30). We also separately set $\gamma_w = 0$ or $\gamma_b = 0$ in order to see which of Σ_w and Σ_b is more reliably estimated. As shown in Figure 3.3, MAP-CVM consistently achieves better results than ML-CVM, especially when there are only very limited training data (e.g. $r = 25$). And as the size of training set grows, the performance by ML gradually approaches that by MAP. We also observe that setting $\gamma_w = 0$

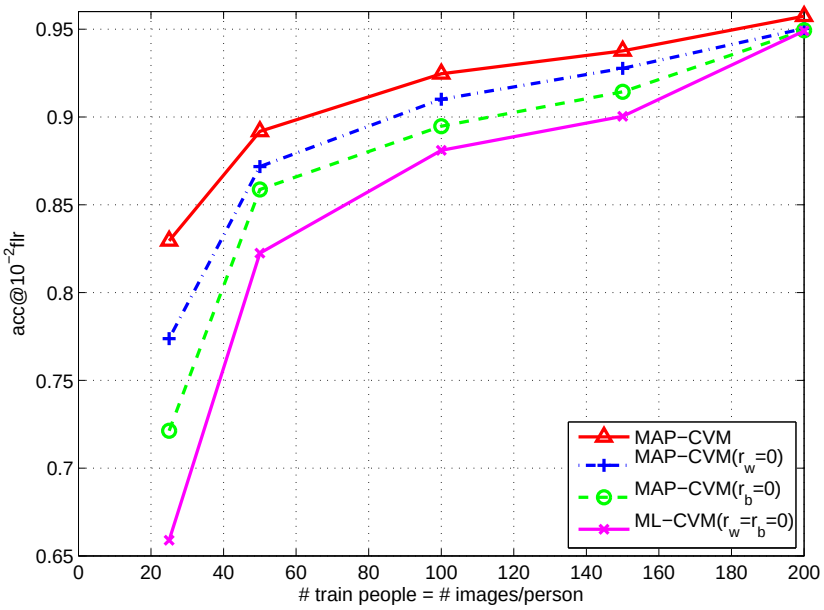


Figure 3.3: Comparison of learning parameters in MAP-CVM.

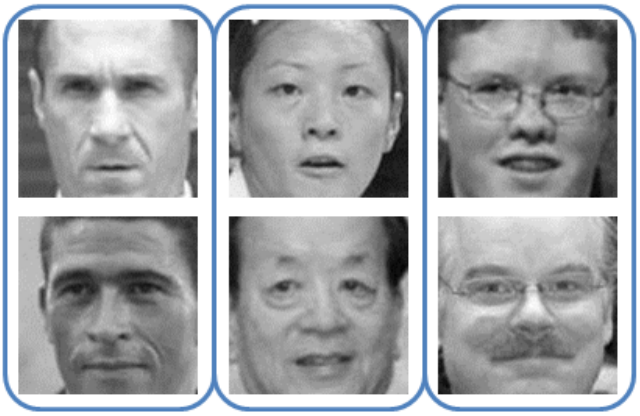
in MAP-CVM results in less degradation of performance than setting $\gamma_b = 0$. This phenomenon to some extent verifies our conjecture that the between-class covariance matrix is more vulnerable to the sparsity of the training data. Note that in the experiment, we set the other prior parameters as $\mu_0 = 0, \Psi_b = \Psi_w = I$, and we set $\gamma_\mu = 0$ since we find the estimation of μ is usually quite robust.

3.2.2 LFW

The "Labeled Faces in the Wild" (LFW) [21] is a database of face images designed for studying the problem of unconstrained face recognition. The face images are downloaded from Yahoo! News in 2002–2003, and demonstrate a large variety of pose, expression, lighting, etc. The dataset contains more than 13,000 face images from 5,749 people, among which 1,680 people have two or more distinct photos. We extract the same HG feature as in previous section. The features are reduced to 500 dimensions by PCA.

We test our algorithm under the standard "image restricted" setting that is particularly designed for verification. In this setting, the dataset is divided into 10 fully independent folds, and it is ensured that the same person does not appear across different folds. Each fold contains between 527 and 609 people, and the number of corresponding faces varies from 1016 to 1783. Moreover, the identities

(a) Example of positive image pairs.



(b) Example of negative image pairs.

Figure 3.4: Example of positive (same-person) and negative (different-person) pairs in the LFW dataset.

of the people are hidden from use. Instead, 300 positive and 300 negative image pairs are provided within each fold. Figure 3.4 shows some examples of positive and negative image pairs. Each time we learn both the PCA projection and the parameters of our decision function on 9 training folds, and evaluate on the remaining fold. Pairwise classification accuracy averaged over 10 runs is reported, as suggested by [21].

As shown in Table 3.2, our approach significantly outperforms state-of-the-art works on the LFW dataset. It should be noted that our verification accuracy of 91.4% is the best reported result[2] on LFW under the category of "no outside data is used beyond alignment/feature extraction."

[2]http://vis-www.cs.umass.edu/lfw/results.html

Table 3.2: Comparison with state-of-the-art algorithms on LFW dataset. The best performance is highlighted in bold.

Methods	Accuracy (%)
MERL+Nowak [26]	76.2
LDML [7]	79.3
LBP + CSML [8]	85.6
CSML + SVM [8]	88.0
Combined b/g samples [27]	86.8
DML-eig combined [28]	85.7
MAP-CVM	**91.4**

CHAPTER 4

DISCRIMINATIVE LEARNING OF DECISION FUNCTIONS

This chapter studies discriminative methods for the verification problem. In particular, we propose to learn a **decision function** in (1.2) directly without having to assume certain probabilistic distributions of the images.

We are motivated from two aspects.

- Traditional discriminative approaches, in particular metric learning (ML), provide a nice framework of learning from pairwise constraints. However, as we will show in Section 4.1, the constant thresholding rule by ML (2.3) is usually suboptimal. We can therefore modify it by adjusting the decision rule locally, i.e., consider $f(x, y) = d(x, y) - (x - y)^T M(x - y)$ where $d(x, y)$ is a function of x, y rather than a constant. As a starting point, we assume $d(x, y)$ takes a simple quadratic form, which leads to our general **second-order decision function**.

- In addition to ML (2.3), the decision functions (or more precisely log-likelihood ratio) by both Bayesian Face Recognition (BFR) (2.2) and the Covariance Verification Model (CVM) (3.8) fall into a second-order form. One may naturally wonder whether it is possible to directly learn a second-order decision function without having to assume certain probabilistic models as in BFR and CVM.

4.1 Bridging Distance Metric and Local Decision Rules

Metric learning (related to feature selection, dimension reduction, or subspace projection, etc.) plays a fundamental role in machine learning. It is particularly important for computer vision applications, where the feature representation of images or videos is usually of complex high-dimensional form [1, 29]. In these

cases, the Euclidean norm associated with the original feature space usually does not provide much useful information for the subsequent learning tasks. In most applications we consider here, the sample data are sparse in the high-dimensional feature space. So we focus on metric learning with respect to a global metric, i.e., the matrix M in (2.3), although learning a local metric has attracted an increasing interest in machine learning research.

However, metric learning itself is insufficient for the verification problem, as discussed in Chapter 2. The problem is that after metric learning, we still need to make a decision. A simple constant threshold in (2.3) is sub-optimal, as shown in Proposition 4.1, even if the associated metric is correct. A decision rule that can adapt to the local structures of data [30] is the key to achieve good verification performance. To this end, we consider a joint model that bridges a global distance metric and a local decision rule, and we further show the optimality of our method over ML in the verification setting.

Consider $f(x, y) = d(x, y) - (x - y)^T M(x - y)$ where $d(x, y)$ acts as a **local decision rule** for a learned metric M. Since the metric itself is quadratic, as a starting point, we also assume $d(x, y)$ takes a simple quadratic form. We will see later in Section 4.2 that this formulation leads to a kernelized large-margin learning problem, and thus can be easily generalized to decision functions of high-orders by the kernel trick [31].

For now, let us focus on the second-order decision rule, i.e., $d(x, y) = \frac{1}{2} z^T Q z + w^T z + b$, where $z^T = \begin{bmatrix} x^T & y^T \end{bmatrix} \in \mathbb{R}^{2d}$, $Q = \begin{bmatrix} Q_{xx} & Q_{xy} \\ Q_{yx} & Q_{yy} \end{bmatrix} \in \mathbb{R}^{2d \times 2d}$, $w^T = \begin{bmatrix} w_x^T & w_y^T \end{bmatrix} \in \mathbb{R}^{2d}$, and $b \in \mathbb{R}$. Due to the symmetry property with respect to x and y, we can rewrite $d(x, y)$ as follows:

$$\begin{aligned} d(x, y) &= \frac{1}{2} x^T \tilde{A} x + \frac{1}{2} y^T \tilde{A} y + x^T \tilde{B} y + c^T (x + y) + b \\ &= \frac{1}{4} (x - y)^T (\tilde{A} - \tilde{B})(x - y) + \frac{1}{4} (x + y)^T (\tilde{A} + \tilde{B})(x + y) \\ &\quad + c^T (x + y) + b, \end{aligned} \tag{4.1}$$

where $\tilde{A} = Q_{xx} = Q_{yy}$ and $\tilde{B} = Q_{xy} = Q_{yx}$ are both $d \times d$ real symmetric matrices (not necessarily positive semidefinite), $c = w_x = w_y$ is a d-dimensional vector, and b is the bias term.

Now we obtain the **second-order decision function** for verification:

$$\begin{aligned} f(x,y) &= d(x,y) - (x-y)^T M (x-y) \\ &= \frac{1}{2}x^T A x + \frac{1}{2}y^T A y + x^T B y + c^T(x+y) + b, \end{aligned} \tag{4.2}$$

by letting $A - B = \tilde{A} - \tilde{B} - 4M$ and $A + B = \tilde{A} + \tilde{B}$. Again, A and B are real symmetric and need not to be PSD.

The above decision function has the following desirable properties:

- **Learning globally, acting locally.** We bridge a global metric M and a local decision rule using a joint model (4.2). Interestingly, the number of parameters is at the same order ($O(d^2)$) as that of ML.
- **Fully informed decision making.** The local decision rule in (4.1) depends not only on $x - y$, the difference vector usually considered by ML, but also on $x + y$, which contains orthogonal information of (x, y) that would otherwise be neglected by $x - y$ alone.
- **Kernelizable to higher order.** As we will see in Section 4.2, the decision function in (4.2) leads to a kernelized large-margin learning problem, and thus can be easily generalized to decision functions of higher-orders by the kernel trick [31].

We now formally show the optimality of our decision function over ML by considering a simple case where two categories of samples in $\mathbb{R}^d$ are linearly separable. We show that in the verification setting, the performance of any given metric is inferior to that of our model, in this simple case.

Proposition 4.1 *Given two linearly separable classes, the verification error rate by our second-order decision function* (4.2) *is always lower than that by a learned metric with a fixed threshold* (2.3)*. More specifically, in this particular setting, our model can always achieve zero verification error while ML does not.*

Proof Suppose the two classes in $\mathbb{R}^d$ satisfy: $w^T x + b > 0$ for class 1, and $w^T x + b < 0$ for class 2. In verification, we aim to identify if two samples x and y are from the same class or different ones.

We first show that our decision function in (4.2) always achieves zero verification error. x and y are from the same class if and only if $(w^T x + b)$ and $(w^T y + b)$ are of the same sign. In other words, we can perfectly identify pairs from the same class vs. those from different classes, by checking the sign of

$(w^T x + b)(w^T y + b) = x^T(ww^T)y + bw^T(x + y) + b^2$. This decision function is clearly a special case of (4.2).

We then show that the ML approach in (2.3) does not always achieve zero verification error. Any Mahalanobis distance between x and y can be regarded as the Euclidean distance on the space transformed by L, namely, $d(x, y) = (x - y)^T M(x - y) = (x - y)L^T L(x - y) = \|x' - y'\|_2^2$, where $M = L^T L$, $x' = Lx$ and $y' = Ly$. In this new space, the two classes are still linearly separable, since $w^T x + b = w'^T x' + b$ and $w' = wL^{-1}$ (assuming M is full rank). Therefore, in order for ML method in (2.3), or simply $|x - y| < d$, to achieve zero verification error, the following condition needs to be satisfied:

$$\max_{m(x)=m(y)} \|x' - y'\|_2 < \min_{m(x)\neq m(y)} \|x' - y'\|_2. \tag{4.3}$$

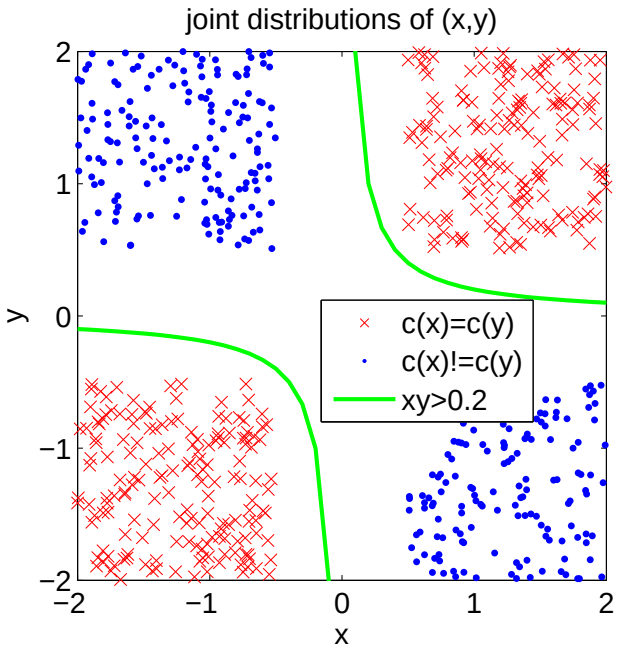
Unfortunately, the above condition does not always hold. Consider a counterexample in 1-D: class 1 is uniformly distributed in $[-2,\ -0.5]$ and class 2 in $[0.5,\ 2]$. The two classes are indeed separable, but condition (4.3) is not satisfied since $\max_{m(x)=m(y)} \|x' - y'\|_2 = 1.5$ and $\min_{m(x)\neq m(y)} \|x' - y'\|_2 = 1$. In fact, from Figure 4.1(b) we see that ML method ($|x-y| < d$) inevitably results in finite verification error, while our model is able to perfectly separate the two types of pairs on the (x, y) space, shown in Figure 4.1(a).

A more realistic example that also violates (4.3) is this: Face images of the same person but from different poses are usually more dissimilar than those from different people but of the same pose. Figure 4.2 shows such an example with selected image pairs of the LFW dataset [21]. ■

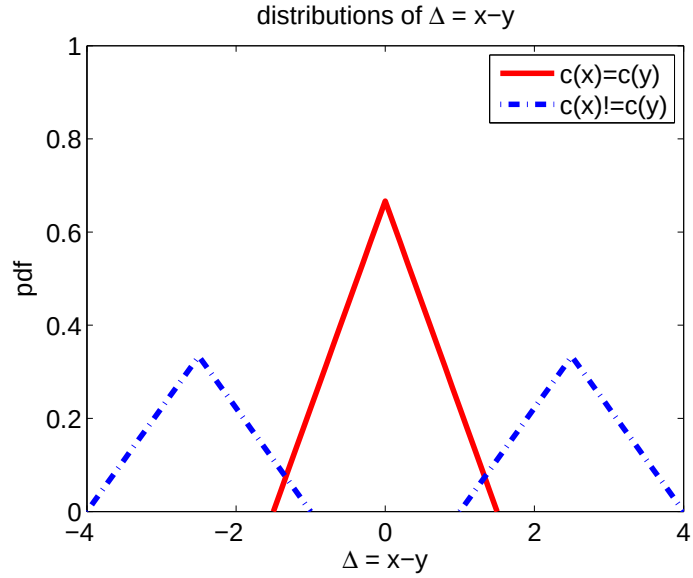
4.2 A Large-Margin Solution with an Efficient Algorithm

4.2.1 A large margin formulation

Recall that the objective of a verification problem is to learn a symmetric decision function: $f(x, y) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ that takes a pair of samples $x, y \in \mathbb{R}^d$ as inputs,



(a) Joint distribution of (x,y), with zero-error decision function by our model: $xy - 0.2 > 0$.



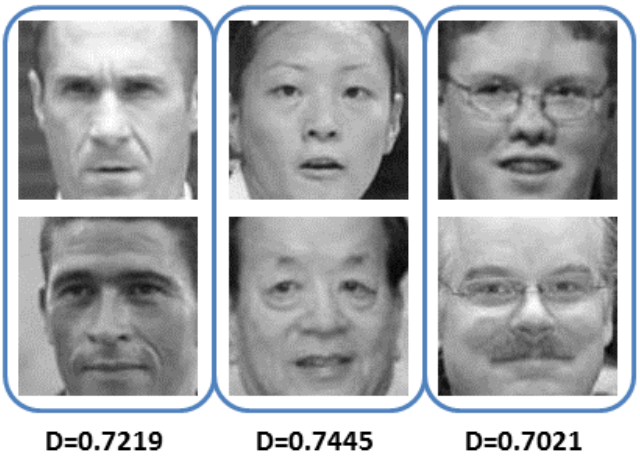
(b) Distribution of $\Delta = x - y$ in case of metric learning, with finite verification error.

Figure 4.1: Distributions of same-class pairs (red) vs. different-class pairs (blue).

with decision rule:

$$f(x,y) \begin{cases} > 0, & \text{if } m(x) = m(y); \\ < 0, & \text{otherwise.} \end{cases} \tag{4.4}$$

(a) Intra-person distances (different poses).



(b) Inter-person distances (same pose).

Figure 4.2: Comparison of intra-person and inter-person distances under a learned metric.

Our goal is to find the optimal second-order decision function $f(x, y)$ in (4.2) that is parametrized by $\{A, B, c, b\}$. This naturally leads to a choice of an SVM-like [3] objective function, as the resulting large-margin model generalizes well to unseen examples.

Specifically, assume we are given a dataset of examples, and pairwise labels are assigned. A sample pair $p_i = (x_i, y_i)$ is labeled as either "positive" ($l_i = +1$), if x_i and y_i are from the same class, or "negative" ($l_i = -1$), otherwise. We further denote by $\mathcal{P}$ the set of all labeled sample pairs. An SVM-like objective function can be formulated as:

$$\begin{aligned} \min \quad & \frac{1}{2}\left(\|A\|_F^2 + \|B\|_F^2 + \|c\|_2^2\right) + \lambda \sum_{i \in \mathcal{P}} \xi_i \\ \text{s.t.} \quad & l_i f(x_i, y_i) \geq 1 - \xi_i \quad \forall i \in \mathcal{P} \\ & \xi_i \geq 0 \quad \forall i \in \mathcal{P}. \end{aligned} \tag{4.5}$$

Here $\|A\|_F = \sqrt{\mathbf{tr}(A^T A)}$ is the Frobenius matrix norm, and $\mathbf{tr}(A)$ denotes the trace of matrix A.

Noticing the inner product defined on the matrix space, $\langle A, B \rangle = \mathbf{tr}(A^T B)$, we reformulate the decision function (4.2) into:

$$\begin{aligned} f(x, y) &= \frac{1}{2}\mathbf{tr}\left(A\left(xx^T + yy^T\right)\right) + \frac{1}{2}\mathbf{tr}\left(B\left(xy^T + yx^T\right)\right) + c^T(x + y) + b \\ &= \frac{1}{2}\left\langle A,\ xx^T + yy^T \right\rangle + \frac{1}{2}\left\langle B,\ xy^T + yx^T \right\rangle + \langle c,\ x + y \rangle + b \\ &= \langle \zeta, \psi(x, y) \rangle + b, \end{aligned} \tag{4.6}$$

where $\zeta \in \mathbb{R}^{2d^2+2}$ is a vectorized representation of the hyper-parameters (excluding b), and $\psi(x, y)$ defines a mapping $\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{2d^2+d}$:

$$\zeta = \begin{bmatrix} \mathbf{vec}(A) \\ \mathbf{vec}(B) \\ c \end{bmatrix}, \qquad \psi(x, y) = \begin{bmatrix} \frac{1}{2}\mathbf{vec}(xx^T + yy^T) \\ \frac{1}{2}\mathbf{vec}(xy^T + yx^T) \\ x + y \end{bmatrix}, \tag{4.7}$$

where $\mathbf{vec}(\cdot)$ denotes the vectorization of a matrix. Note that $\psi(x, y)$ can be viewed as a symmetrization of the original feature space (x, y); that is, any function of $\psi(x, y)$ is now a symmetric function of x and y.

Similarly, the objective function can be rewritten as:

$$\begin{aligned} \min \quad & \frac{1}{2}\langle\zeta,\zeta\rangle + \lambda\sum_{i\in\mathcal{P}}\xi_i \\ \text{s.t.} \quad & l_i(\langle\zeta,\psi_i\rangle + b) \geq 1-\xi_i \quad \forall i\in\mathcal{P} \\ & \xi_i \geq 0 \quad \forall i\in\mathcal{P}, \end{aligned} \tag{4.8}$$

where ψ_i is an abbreviation of $\psi(x_i, y_i)$. This looks identical to the standard SVM problem [3]. Thus existing SVM solvers could be employed to solve this problem, such as stochastic gradient decent [32] that works on the primal problem directly, or sequential minimal optimization (SMO) [33] that solves the dual problem instead.

4.2.2 An efficient dual solver

Though it appears straightforward, solving (4.8) directly is infeasible due to the high dimensionality of $2d^2 + d$. For instance, a moderate image feature of 1000 dimensions will lead to more than 1 million parameters to estimate. What's more, direct application of existing SVM solvers may require forming $\psi(x_i, y_i)$'s explicitly, which is highly inefficient and prohibitive in memory usage. In this section, we will show that the original problem can actually be converted into a kernelized SVM problem that could be solved much more efficiently.

We start with the Lagrange dual of (4.8):

$$\begin{aligned} \max \quad & \frac{1}{2}\sum_i \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j l_i l_j \langle\psi_i,\psi_j\rangle \\ \text{s.t.} \quad & \textstyle\sum_i \alpha_i l_i = 0 \\ & 0 \leq \alpha_i \leq \lambda, \end{aligned} \tag{4.9}$$

where α_i is the Lagrange multiplier corresponding to the i-th constraint. If we could have solved the above problem with optimal α_i^*'s, the solution for the primal is then given by:

$$\zeta^* = \sum_i \alpha_i^* l_i \psi_i \tag{4.10}$$

$$b^* = -l_i - \langle\zeta^*,\psi_i\rangle, \quad \forall i : 0 < \alpha_i^* < \lambda. \tag{4.11}$$

And the optimal decision function is therefore:

$$f(x,y) = \langle \zeta^*, \psi(x,y) \rangle + b^* = \sum_i \alpha_i^* l_i \langle \psi_i, \psi \rangle + b^*. \tag{4.12}$$

We notice that either solving the dual problem (4.9) or applying the optimal function (4.12) involves only the so-called kernel function $K(\psi_i, \psi_j) = \langle \psi_i, \psi_j \rangle$. By substituting (4.7) and the equality $\mathbf{vec}(A)^T \mathbf{vec}(B) = \langle A, B \rangle = \mathbf{tr}(A^T B)$, we arrive at:

$$\begin{aligned} K(\psi_i, \psi_j) &= \frac{1}{4}\mathbf{tr}\left((x_i x_i^T + y_i y_i^T)(x_j x_j^T + y_j y_j^T)\right) \\ &\quad + \frac{1}{4}\mathbf{tr}\left((x_i y_i^T + y_i x_i^T)(x_j y_j^T + y_j x_j^T)\right) + (x_i + y_i)^T (x_j + y_j) \\ &= \frac{1}{4}(x_i^T x_j + y_i^T y_j)^2 + \frac{1}{4}(x_i^T y_j + y_i^T x_j)^2 + (x_i + y_i)^T (x_j + y_j). \end{aligned} \tag{4.13}$$

Note that the kernel function here is defined on a new space of $\psi(x, y)$ that is symmetric with respect to x and y. More specifically, in contrast to the traditional kernel function that is between two individual samples, $K(\psi_i, \psi_j)$ is defined between two pairs of samples.

We now see that, to evaluate each kernel function $K(\psi_i, \psi_j)$, one only needs to calculate 4 inner products on $\mathbb{R}^d$: $x_i^T x_j$, $x_i^T y_j$, $y_i^T x_j$, and $y_i^T y_j$, rather than working on the $(2d^2 + d)$-dimensional space instead. In this way we reduce the complexity of each kernel evaluation from $O(d^2)$ to $O(d)$, which is usually the most costly operation in solving large-scale dual SVM problems [34]. In addition, the memory cost is alleviated accordingly, as explicitly constructing $\psi(x, y)$'s by (4.7) is no longer necessary. Based on (4.13), existing dual SVM solvers such as SMO algorithm [33, 34] can be applied to solve (4.9) efficiently.

Moreover, the fact that only inner products are involved in $K(\psi_i, \psi_j)$ implies the extension to implicit kernel embedding of original features, namely,

$$\begin{aligned} K(\psi_i, \psi_j) &= \frac{1}{4}(G(x_i, x_j) + G(y_i, y_j))^2 + \frac{1}{4}(G(x_i, y_j) + G(y_i, x_j))^2 \\ &\quad + G(x_i, x_j) + G(x_i, y_j) + G(y_i, x_j) + G(y_i, y_j), \end{aligned} \tag{4.14}$$

where $G(\cdot, \cdot)$ is a kernel function of the original feature space. Based on this kernel embedding, we can thus extend our decision function (4.2) to higher orders by the kernel trick [31]. However, in practice, cubic polynomials or higher or-

der functions often work less robustly, so in experiments, we will mainly use the second-order decision functions.

4.3 Experiments

We conduct experiments on three different datasets: Viewpoint Invariant Pedestrian Recognition (VIPeR) [11], Context Aware Vision using Image-based Active Recognition for Re-Identification (CAVIAR4REID) [35], and Labeled Faces in the Wild (LFW) [21]. The first two datasets focus on person verification from human body images, and the latter on face verification. In each experiment, we present results by comparing with classic metric learning (ML) algorithms as well as other state-of-the-art approaches. We demonstrate that our proposed approach significantly outperforms existing works and achieves state-of-the-art results on all datasets.

4.3.1 VIPeR

The VIPeR dataset consists of images from 632 pedestrians with resolution 48×128. For each person, a pair of images are taken from different cameras with widely view angle variations. Viewpoint change of 90 degree or more as well as huge lighting variations make this dataset one of the most challenging datasets available for human body verification. Example images are shown in Figure 4.3.

As in Section 3.2, we extract HG features except that here we use local color descriptors instead of SIFT. To accelerate the learning process, we further reduce the dimensionality of the final feature representation to 600 using PCA (learned on the training set). We also follow exactly the same setup as in [36, 11, 35]: each time half of the 632 people are selected randomly to form the training set, and the remaining people are left for testing (so that no people will appear in both the training and testing). The cumulative matching characteristic (CMC) curve, an estimate of the expectation of finding the correct match in the top n matches, is calculated on the testing set to measure the verification performance (see [36] for details on computing the CMC curve). The final results are averaged over ten random runs.

Figure 4.4(a) compares our proposed method with classic ML algorithms, LMNN [37] and ITML [38], using the same feature. It is apparent that, in the verification

Figure 4.3: Example images of VIPer dataset. Each column shows two images of the same pedestrian captured under two different cameras.

problem, the optimal second-order decision function (4.2) does significantly improve over traditional ML approaches with a fixed threshold (2.3). Note that here LMNN performs the worst. One possible reason is that each class contains only two examples with huge intra-class variations. We are also interested in comparing with other state-of-the-art methods on this dataset, though different features and/or learning algorithms have been used. Figure 4.4(b) shows the comparison with PS [35], SDALF [39], ELF [11], and PRSVM [40]. Clearly our method outperforms all previous works and achieves state-of-the-art performance.

4.3.2 CAVIAR4REID

CAVIAR4REID [35], extracted from the CAVIAR dataset, is another famous dataset widely used for person verification tasks. This dataset not only covers a wide range of poses and real surveillance footage, but also includes multiple images per pedestrian with different view angles and resolutions. There are in total 72 pedestrians, and each person has images recorded from two different cameras in an indoor shopping mall in Lisbon. All the human body images have been cropped with respect to the ground truth, and the resolution varies from 17×39 to 72×144. Here we extract the same image feature as in Section 4.3.1, and we also use the same training/testing protocol.

Again, we compare with popular ML algorithms as well as other state-of-the-art approaches, as shown in Figure 4.5(a) and Figure 4.5(b), respectively. As in Section 4.3.1, we observe a substantial improvement over traditional ML algorithms, and our method also outperforms state-of-the-art works including PS [35] and SDALF [39]. It should be noted that the curves by both PS and SDALF shown in 4.5(b) have been extrapolated for the sake of fair comparison. The reason is that we have to separate a subset of 36 people for learning the parameters of our decision function (4.2) or distance metric. With only half of the people left in testing, we rescale the horizontal axis of PS and SDALF by 50% for a fair comparison.

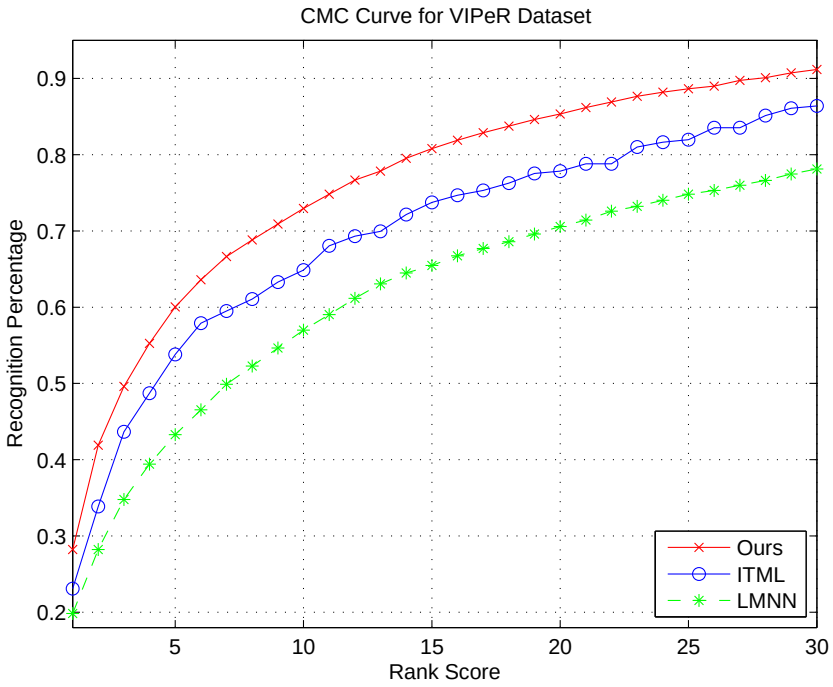
4.3.3 LFW

Again we conduct experiments on the LFW dataset, under exactly the same setting as in Section 3.2.2. Table 4.1 shows that our approach significantly outperforms state-of-the-art works on the LFW dataset. It should be noted that our verification accuracy of 89.6% is the best reported result[1] on LFW under the category of "no outside data is used beyond alignment/feature extraction."

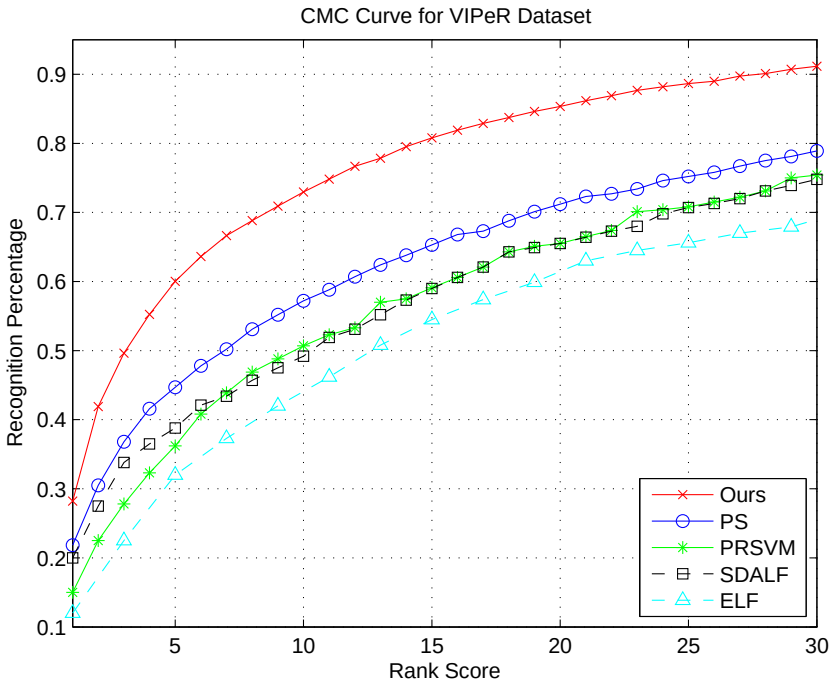
Table 4.1: Comparison with state-of-the-art algorithms on LFW dataset. The best performance is highlighted in bold.

Methods	Accuracy (%)
MERL+Nowak [26]	76.2
LDML [7]	79.3
LBP + CSML [8]	85.6
CSML + SVM [8]	88.0
Combined b/g samples [27]	86.8
DML-eig combined [28]	85.7
Our method	**89.6**

[1] http://vis-www.cs.umass.edu/lfw/results.html

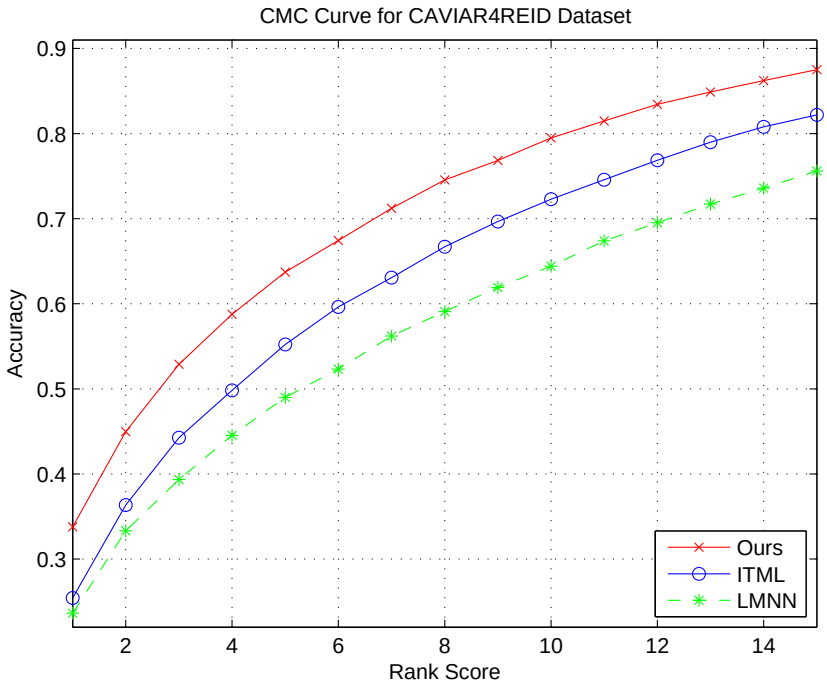


(a) Comparison with metric learning algorithms.

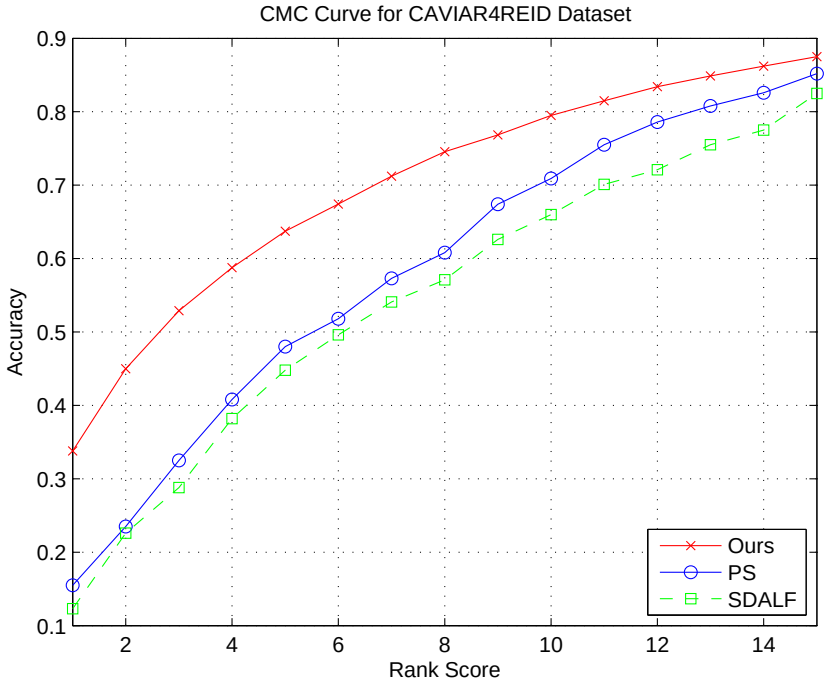


(b) Comparison with other state-of-art algorithms.

Figure 4.4: Experimental results on VIPeR dataset.



(a) Comparison with metric learning algorithms.



(b) Comparison with other state-of-art algorithms. Note that we have rescaled the curves by PS [35] and SDALF [39] for a fair comparison. See text.

Figure 4.5: Experimental results on CAVIAR4REID dataset.

CHAPTER 5

MULTI-SETTING PERSON VERIFICATION

Chapter 3 and Chapter 4 study models for person verification in general. The goal of these models is mainly to resolve the limited-sample issue in person identification systems. In this chapter, we investigate the other challenge in person verification – environment mismatch, which often hugely degrades the performance of conventional approaches in practical applications.

In this chapter we propose a **Multi-setting Covariance Verification Model (MCVM)** for this purpose. It generalizes the Covariance Verification Model (CVM) in Chapter 3 by modeling the environmental variations of images. Experimental results show significant improvement over a universal CVM when environment mismatch exists.

5.1 Environment Mismatch in Person Verification

Human images often exhibit huge variations under different imaging *settings* or *environments*, including viewpoint, lighting conditions, occlusion, image quality, etc. Examples of these variations are shown in Figure 1.1 and Figure 3.1. These variations pose a tremendous challenge to person verification in real-world application scenarios. One direct consequence is the so-called "environment mismatch" issue: the probe/query image is acquired under a different setting from images in the gallery set. In other words, we need to deal with the cross-setting person verification problem, i.e., to identity if two images that are captured under different environments are from the same person.

Typical examples of environment mismatch include tracing a suspect in the surveillance systems, with only few ID photos available to the police. This inevitably involves comparing an ID photo with images captured under surveillance scenarios. While the former is usually captured with a high-quality camera, under frontal pose and sufficient lighting, and without facial expression or occlusions,

the latter could look totally different, with uncontrolled outdoor lighting, large viewpoint deviation from the frontal pose, occasional facial movement, and even substantial motion blurs.

The underlying cause for the failure of traditional approaches in mismatches cases is the multi-modality of image distributions. Given the complexity of various imaging environments, it is more natural to assume that image distribution exhibits multi-mode property. In contrast, most existing person identification or verification methods, not only simple ones like eigen-face [41] and Fisher-face [42], but also our models proposed in previous chapters, consider only simple models with uni-mode distributions (e.g. the single Gaussian assumption in CVM). Therefore, we might need to consider multi-mode models for multi-setting person verification.

Before going into details of our approach, we briefly review existing approaches to deal with the environmental mismatch. One simple approach one can always envision is to the "universal" approach, that is, to disregard the environmental differences, and learn a universal model for all settings. As we have discussed above, this would perform poorly with the existence of multiple complex environments. A number of research works focus on how to convert images captured under different settings to a unified one. For instance, [43, 44] resort to appearance or shape models for aligning face images under different viewpoints to the frontal view. However, these methods work primarily for pose mismatch, and no general solution exists for other types of environment mismatch. More recently, learning-based approaches that are supposed to work with general environment mismatch have become popular. Among them a large family of methods tries to synchronize different settings by learning, for each setting, a separate embedding onto a common subspace. These approaches, such as Multiview Fisher Discriminant Analysis (MFDA) [45], Common Discriminant Feature Extraction (CDFE) [46], Multi-set CCA (MCCA) [47], Generalized Discriminant Analysis (GDA) [48], and Multi-view Discriminant Analysis (MvDA) [49], are mostly motivated by Canonical Correlation Analysis (CCA) [50], and consider extra discriminative information (similar as LDA) in learning the subspace embedding. However, none of them are designed for the verification problem rather than person identification.

5.2 Multi-setting Covariance Verification Model (MCVM)

In this section, we propose a **Multi-setting Covariance Verification Model (MCVM)** for person verification across settings. We start with separate CVMs for different settings. Suppose there are K different settings, and each one of them is associated with a simple CVM:

$$x_{ij}^{(k)} = m_i^{(k)} + e_{ij}^{(k)}, \tag{5.1}$$

where $x_{ij}^{(k)}$ denotes the j-th image of i-th person under setting k, whose distribution is given by

$$m^{(k)} \sim \mathcal{N}(\mu^{(k)}, \Sigma_b^{(k)}), \tag{5.2}$$

$$x^{(k)}|m^{(k)} \sim \mathcal{N}(\mu^{(k)} + m^{(k)}, \Sigma_w^{(k)}). \tag{5.3}$$

However, the above formulation provides no solution for verifying image pairs from two settings due to the lack of knowledge about the relationship between the two settings. Therefore, further assumptions need to be imposed. Knowing that under two settings (e.g. setting k and l), the latent identity components of the same person (e.g. $m^{(k)}$ and m^{l}) might exhibit certain correlation, we use a joint Gaussian to model them, with cross-covariance denoted by $\Sigma_b^{(kl)} = \text{Cov}(m^{(k)}, m^{(l)})$. More generally, we replace the K independent Gaussians in (5.2) by

$$\begin{bmatrix} m^{(1)} \\ m^{(2)} \\ \vdots \\ m^{(K)} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu^{(1)} \\ \mu^{(2)} \\ \vdots \\ \mu^{(K)} \end{bmatrix}, \begin{bmatrix} \Sigma_b^{(11)} & \Sigma_b^{(12)} & \cdots & \Sigma_b^{(1K)} \\ \Sigma_b^{(21)} & \Sigma_b^{(22)} & \cdots & \Sigma_b^{(2K)} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_b^{(K1)} & \Sigma_b^{(K2)} & \cdots & \Sigma_b^{(KK)} \end{bmatrix} \right) \tag{5.4}$$

where $\Sigma_b^{(kk)} = \Sigma_b^{(k)}, \ \forall k$. We term this model **Multi-setting Covariance Verification Model**, or **MCVM** in short.

Note that the parameters of MCVM fall into two groups:

- Intra-setting parameters: $\boldsymbol{\theta}_I = \{\mu^{(k)}, \ \Sigma_w^{(k)}, \ \Sigma_b^{(k)}, \ \text{for } 1 \leq k \leq K\}$;
- Inter-setting parameters: $\boldsymbol{\theta}_E = \{\Sigma_b^{(kl)}, \ \text{for } 1 \leq k, l \leq K \text{ and } k \neq l\}$.

Essentially, $\boldsymbol{\theta}_I$ models the image distribution under each setting with a separate

CVM, and $\boldsymbol{\theta}_E$ associates different settings by modeling the correlations of the identity components.

Before going into the details of the learning and evaluation of MCVM, we summarize its merits as follows:

- **Unified model for multi-setting person verification.** Many of the previous multi-setting approaches only consider a special type of environment mismatch (e.g. pose), or provide solutions for two settings only (e.g. CCA [50] and CDFE [46]). Moreover, most existing approaches are designed for person identification rather than verification.

- **Simple (closed-form) estimation algorithm.** Similar to CVM, we derive a closed-form solution for learning the parameters of MCVM, with only simple linear algebra operations. This solution, though not exact, yields comparable results to the more complicated EM algorithm. In contrast, existing approaches often involve numerical optimization of complicated objective functions, or solving a large eigen decomposition problem.

- **MAP estimation for insufficient training data.** We estimate the parameters of MCVM under maximum a posteriori (MAP) criterion. This is particularly useful for estimating the between-class covariances matrices (both intra-setting and inter-setting) in (5.4), due to the insufficient number of people in training.

- **Easy data collection.** Our model correlates different settings through the identity components which appear at the person level. Thus we only need to guarantee that each person be captured under different settings (intrapersonal variations could be arbitrary). In contrast, some existing works model image-level correlations, and thus require the same intrapersonal variations for all people under all settings, which makes the data collection much more difficult and expensive.

5.2.1 Scoring of MCVM

Given a learned model, the scoring of MCVM is slightly more complicated than that of CVM (3.8). Given two test images, $x_1^{(k)}$ and $x_2^{(l)}$ from setting k and l

respectively, the log-likelihood ratio by MCVM can be computed as:

$$LLR(x_1^{(k)}, x_2^{(l)}) = \frac{1}{2} x_1'^{(k)T} A_k^{(kl)} x_1'^{(k)} + \frac{1}{2} x_2'^{(l)T} A_l^{(kl)} x_2'^{(l)} + x_1'^{(k)T} B^{(kl)} x_2'^{(l)} + c^{(kl)} \tag{5.5}$$

where $x_1'^{(k)} = x_1^{(k)} - \mu^{(k)}$, $x_2'^{(l)} = x_2^{(l)} - \mu^{(l)}$, and

$$\begin{aligned}
A_k^{(kl)} &= -{\Sigma_t^{(k)}}^{-1} \Sigma_b^{(kl)} {M^{(lk)}}^{-1} \Sigma_b^{(lk)} {\Sigma_t^{(l)}}^{(-1)}, \\
A_l^{(kl)} &= {\Sigma_t^{(l)}}^{-1} - {M^{(lk)}}^{-1}, \\
B^{(kl)} &= {\Sigma_t^{(k)}}^{-1} \Sigma_b^{(kl)} {M^{(lk)}}^{-1}, \\
c^{(kl)} &= \log \left| \Sigma_t^{(l)} \right| - \log \left| M^{(kl)} \right|, \\
M^{(lk)} &= \Sigma_t^{(l)} - \Sigma_b^{(lk)} {\Sigma_t^{(k)}}^{-1} \Sigma_b^{(kl)}.
\end{aligned}$$

We omit the proof here, which follows from similar derivation in Section 3.1.1.

5.2.2 Closed-form solution for MCVM

The exact inference of MCVM parameters under either ML or MAP criterion can be carried out by the EM algorithm. Alternatively, we could obtain approximate closed-form solution, which proves to be much more efficient and perform similarly as the EM algorithm. Therefore, we only present the latter method here.

Under mild assumptions (a moderate number of examples per person), the hidden identity components can be regarded as if they were exactly inferred from the empirical means, i.e.,

$$m_i^{(k)} \approx \tilde{\mu}_i^{(k)} = \frac{1}{n_i^{(k)}} \sum_{j=1}^{n_i^{(k)}} x_{ij}^{(k)}, \ \ \forall k \text{ and } i,$$

where $n_i^{(k)}$ is the number of images of the i-th person under setting k. Then we

arrive at the following empirical estimation under ML criterion:

$$\begin{aligned}
\tilde{\mu}^{(k)} &= \frac{1}{n^{(k)}} \sum_{i=1}^{r} n_i^{(k)} \tilde{\mu}_i^{(k)}, \ \ \forall k, \\
\tilde{\Sigma}_w^{(k)} &= \frac{1}{n^{(k)}} \sum_{i=1}^{r} \sum_{j=1}^{n_i^{(k)}} (x_{ij}^{(k)} - \tilde{\mu}_i^{(k)})(x_{ij}^{(k)} - \tilde{\mu}_i^{(k)})^T, \ \ \forall k, \\
\tilde{\Sigma}_b^{(k)} &= \frac{1}{n^{(k)}} \sum_{i=1}^{r} n_i^{(k)} (\tilde{\mu}_i^{(k)} - \tilde{\mu}^{(k)})(\tilde{\mu}_i^{(k)} - \tilde{\mu}^{(k)})^T, \ \ \forall k, \\
\tilde{\Sigma}_b^{(kl)} &= \frac{2}{n^{(k)} + n^{(l)}} \sum_{i=1}^{r} \frac{n_i^{(k)} n_i^{(l)}}{n_i^{(k)} + n_i^{(l)}} (\tilde{\mu}_i^{(k)} - \tilde{\mu}^{(k)})(\tilde{\mu}_i^{(l)} - \tilde{\mu}^{(l)})^T, \ \ \forall k \neq j,
\end{aligned}$$

where $n^{(k)} = \sum_{i=1}^{r} n_i^{(k)}$ is the total number of images under setting k, and r is the number of people in the training set.

Similarly the approximate solution for MAP estimation is as follows:

$$\begin{aligned}
\hat{\mu}^{(k)} &= \tilde{\mu}^{(k)}, \ \ \forall k, \\
\hat{\Sigma}_w^{(k)} &= \alpha_w^{(k)} \tilde{\Sigma}_w^{(k)} + (1 - \alpha_w^{(k)}) \Psi_w^{(k)}, \ \ \forall k, \\
\hat{\Sigma}_b^{(k)} &= \alpha_b^{(k)} \tilde{\Sigma}_b^{(k)} + (1 - \alpha_b^{(k)}) \Psi_b^{(k)}, \ \ \forall k, \\
\hat{\Sigma}_b^{(kl)} &= \alpha_b^{(kl)} \tilde{\Sigma}_b^{(kl)} + (1 - \alpha_b^{(kl)}) \Psi_b^{(kl)}, \ \ \forall k \neq l.
\end{aligned}$$

Here $\{\alpha_w^{(k)}\}$, $\{\alpha_b^{(k)}\}$, and $\{\alpha_b^{(kl)}\}$, given in a similar form as (3.30), serve as the smoothing/interpolation coefficients for MAP estimation, while $\{\Psi_w^{(k)}\}$, $\{\Psi_b^{(k)}\}$, $\{\Psi_b^{(k)}\}$ specify the prior parameters for $\{\tilde{\Sigma}_w^{(k)}\}$, $\{\tilde{\Sigma}_b^{(k)}\}$, and $\{\tilde{\Sigma}_b^{(kl)}\}$ respectively (see Section 3.1.2 for more discussion on the prior distributions). In practice, we find that simply setting $\alpha_w^{(k)} = \Psi_w^{(k)} = \Psi_w^{(kl)} = I, \ \forall k, l$ works pretty well, and we determine $\{\alpha_w^{(k)}\}$, $\{\alpha_b^{(k)}\}$, and $\{\alpha_b^{(kl)}\}$ via cross-validation. Note that we again disregard the priors for the $\{\mu^{(k)}\}$, and use the empirical estimates instead.

5.3 Experiments

In this section, we conduct experiments on two datasets. One is the same Multi-PIE subset used in Chapter 3, and the other is the CIGIT-AIS dataset. We compare the proposal MCVM with both universal CVM and other state-of-the-art algorithms for multi-setting person identification (e.g. Multi-view Discriminant Anal-

ysis (MvDA) [49]).

5.3.1 Multi-PIE

In this experiment, we use the same subset of CMU Multi-PIE database as Section 3.2.1, with 5 different views, from -30° to 30°, and 13 lighting conditions. Here we treat the viewpoints as the "settings," and the differences in lighting conditions and facial expressions as intrapersonal noises. For each train/test partition, we fix the training set as consisting of $r = 200$ people (instead of varying r), each with 200 images. This gives us around 40 training images per person under each different view. Again we use 30 people for testing.

Since we are mainly interested in studying the effect of viewpoint mismatches, we purposely set up a "one-gallery-image" identification task (which we have shown in Chapter 2 to be equivalent to verification). In particular, we use the same train/test partition as Section 3.2.1. For testing, we randomly choose one frontal view image from each test person to form a gallery set, and leave the rest as probe images. We then do person identification with the decision rule in (2.1). The average identification/verification rate[1] is reported for each view separately.

Here we compare with three methods shown in Table 5.1. "CVM (frontal)" stands for training a single CVM on the frontal view data only, while "CVM (universal)" learns a universal CVM on images from all poses. "MvDA" or Multiview Discriminant Analysis (MvDA) [49], on the other hand, is a multi-setting person identification (not verification) algorithm proposed recently. It has shown to outperform most previous methods in its category, including Multiview Fisher Discriminant Analysis (MFDA) [45], Common Discriminant Feature Extraction (CDFE) [46], Multi-set CCA (MCCA) [47], and Generalized Discriminant Analysis (GDA) [48].

Table 5.1 reports the average verification rate for each probe view. Clearly our MCVM achieves the best results among all methods. In addition, its performance degrades much slower as the amount of mismatch increases. For example, the accuracy of "CVM (frontal)" is as high as MCVM with frontal probe image (no mismatch), but drops quickly as probe images deviate from frontal pose (large mismatch). In contrast, our MCVM still archives around 82% verification accuracy for $\pm 30°$ probe images. Universal CVM simply disregards the setting

[1] Since (2.1) is essentially based on the verification, we use "verification rate" and "identification rate" interchangeably.

differences and thus performs poorly even with frontal probe image. Our method is also much better than MvDA,[2] though as a multi-view method, MvDA also demonstrates insignificant performance degradation with mismatch. We believe that the moderate performance of MvDA is mainly due to its limitation in person identification problems

Table 5.1: Multi-view verification rate on Multi-PIE dataset. The best performance is highlighted in bold.

Method	-30°	-15°	0°	15°	30°
CVM (frontal)	34.6%	63.3%	88.3%	56.7%	30.7%
CVM (universal)	77.9%	84.4%	82.3%	83.4%	75.3%
MvDA [49]	77.5%	81.8%	83.7%	84.1%	74.1%
MCVM	**82.4%**	**86.5%**	**88.3%**	**88.3%**	**83.3%**

5.3.2 CIGIT-AIS

CIGIT-AIS is a much more challenging dataset for multi-setting person verification. It is collected by Chongqing Institute for Green and Intelligent Technology (CIGIT), Chinese Academy of Sciences. This dataset captures images under three very different scenarios: camera array (A), ID photo (I) and surveillance videos (S), as shown in the middle, left and right columns of Figure 5.1 respectively. We note that not only do the three main settings greatly differ from each other, but also there exist significant variations within the same setting (e.g. pose, lighting condition, occlusion for camera array, and motion blur for surveillance videos). In summary, the CIGIT-AIS dataset consists of 120 subjects. And each of them has around 200 images acquired with camera array, 80 ID photos, and 100 image from surveillance videos.

Here we adopt a similar setup as Section 5.3.1. We select 100 people for training, and 20 remaining ones for gallery and probe. We fix the gallery image to be from ID photos. Verification rate is reported for the three probe settings, as shown in Table 5.2. Again, our MCVM significantly outperforms other methods. We also note that the performance gap between ID photo and surveillance videos is much larger than that between ID photo and camera array. This in some sense implies the great difficulty in applying person verification in surveillance scenarios.

[2]Note that we have tuned the hyper-parameter of MvDA (number of dimensions) for its best performance.

Figure 5.1: Example images from CIGIT-AIS dataset. Each row shows the example images from the same subject. The left three columns show the ID photo, the middle three columns show the images captured with camera array, and the right two columns show images from surveillance videos.

Table 5.2: Multi-setting verification rate on CIGIT-AIS dataset. The best performance is highlighted in bold.

Method	ID photo	Camera array	Surveil. videos
CVM (ID)	88.9%	55.0%	32.3%
CVM (universal)	69.0%	67.8%	50.1%
MvDA [49]	80.6%	66.8%	52.1%
MCVM	**88.9%**	**72.1%**	**60.1%**

CHAPTER 6

LEARNING TOWARD EFFICIENT PERSON SEARCH

In the previous chapters, we develop both generative and discriminative models for the person verification problem. In this chapter, we look into a potential application – *person search*.

Imagine a typical application scenario: we have a picture (e.g. an ID photo) of a target subject, and would like to search within a surveillance system the appearances of this particular person. At first glance, this task looks quite straightforward as it merely involves ranking all candidate images (detected faces or human bodies) in the video recordings with respect to their "identity similarities" to the query image. While we have found meaningful similarity measures such as the log-likelihood ratio in (3.8), and the decision function in (4.2), what remains is computing the similarity function between all candidate images and the query image. This procedure, however, becomes impractical when it comes to the entire surveillance video recordings in a large city such as New York or London, where millions of face or human body images are captured per day. The person search problem becomes even more challenging due to the high dimensionality of image representations (e.g. HG [1]), and the quadratic complexity of evaluating either (3.8) or (4.2).

We abstract the person search problem to *fast similarity search*, a better-understood problem in the computer vision and machine learning communities. The objective is to efficiently obtain the most "similar" items to an input query in a huge database, according to a given similarity measure. This chapter studies this problem in general.

6.1 Background

The design of efficient algorithms for large scale similarity search (such as nearest neighbor search) has been a central problem in computer science. This prob-

lem becomes increasingly challenging in modern applications because the scale of modern databases has grown substantially and many of them are composed of high dimensional data. This means that classical algorithms such as kd-trees are no longer suitable [51] and new algorithms have to be designed to handle high dimensionality. However, existing approaches for large scale search in high dimension relied mainly on algorithmic constructions that are either data independent or weakly dependent. Motivated by the success of machine learning in the design of ranking functions for information retrieval (the *learning to rank* problem [52, 53]) and the design of compact embedding into binary codes (the *learning to hash* problem [54]), it is natural to ask whether we can use machine learning (in particular, supervised learning) to optimize data structures that can improve search efficiency. We call this problem **learning to search**, and this paper demonstrates that supervised learning can lead to improved search efficiency over algorithms that are not optimized using supervised information.

To leverage machine learning techniques, we need to consider a scalable search structure with parameters optimizable using labeled data. The data structure considered in this chapter is motivated by the success of the vocabulary tree method in image retrieval [55, 56, 57], which has been adopted in modern image search engines to find near duplicate images. Although the original proposal was based on "bag of local patch" image representation, this chapter considers a general setting where each database item is represented as a high dimensional vector. Recent advances in computer vision show that it is desirable to represent images as numerical vectors of as high as thousands or even millions of dimensions [58, 1]. We can easily adapt the vocabulary tree to this setting: we partition the high dimensional space into disjoint regions using hierarchical k-means, and regard them as the "vocabulary." This representation can then be integrated into an inverted index based text search engine for efficient large scale retrieval. In this paper, we refer to this approach as *k-means trees* because the underlying algorithm is the same as in [59, 60]. Note that k-means trees can be used for high dimensional data, while the classical kd-trees [61, 62, 63] are limited to dimensions of no more than a few hundred.

In this chapter, we also adopt the tree structural representation, and propose a learning algorithm to construct the trees using supervised data. It is worth noting that the k-means trees approach suffers from several drawbacks that can be addressed in our approach. First the k-means trees only use unsupervised clustering algorithm, which is not optimized for search purposes; as we will show

in the experiments, by employing supervised information, our learning to search approach can achieve significantly better performance. Second, the underlying k-means clustering limits the k-means tree approach to Euclidean similarity measures (though possible to extended to Bregman distances), while our approach can be easily applied to more general metrics (including semantic ones) that prove effective in many scenarios [64, 65, 66]. Nevertheless our experiments still focus on Euclidean distance search, which is to show the advantage over the k-means trees.

The learning to search framework proposed in this chapter is based on a formulation of search as a supervised learning problem that jointly optimizes two key factors of search: *retrieval quality* and *computational cost*. Specifically, we learn a set of *selection functions* in the form of a tree ensemble, as motivated by the aforementioned kd-trees and k-means trees approaches. However, unlike the traditional methods that are based only on unsupervised information, our trees are learned under the supervision of pairwise similarity information, and are optimized for the defined search criteria, i.e., to maximize the retrieval quality while keeping the computational cost low. In order to form the forest, boosting is employed to learn the trees sequentially. We call this particular method **Boosted Search Forest** (BSF).

6.1.1 Related work

It is worth comparing the influential Locality Sensitive Hashing (LSH) [67, 68] approach with our learning to search approach. The idea of LSH is to employ random projections to approximate the Euclidean distance of original features. An inverted index structure can be constructed based on the hashing results [67], which facilitates efficient search. However, the LSH algorithm is completely data independent (using random projections), and thus the data structure is constructed without any learning. While interesting theoretical results can be obtained for LSH, as we shall see with the experiments, in practice its performance is inferior to the data-dependent search structures optimized via the learning to search approach of this chapter.

Another closely related problem is *learning to hash*, which includes Boost-SSC [69], Spectral Hashing [70], Restricted Boltzmann Machines [71], Semi-Supervised Hashing [72], Hashing with Graphs [73], etc. However, the motiva-

tion of the hashing problem is fundamentally different from that of the search problem considered in this work. Specifically, the goal of learning to hash is to embed data into compact binary codes so that the Hamming distance between two codes reflects their original similarity. In order to perform efficient Hamming distance search using the embedded representation, an additional efficient algorithmic structure is still needed. (How to come up with such an efficient algorithm is an issue usually ignored by learning to hash algorithms.) The compact hash codes were traditionally believed to achieve low search latency by employing either linear scan, hash table lookup, or more sophisticated search mechanism. As we shall see in our experiments, however, linear scan on the Hamming space is not a feasible solution for large scale search problems. Moreover, if other search data structure is implemented on top of the hash code, the optimality of the embedding is likely to be lost, which usually yields suboptimal solution inferior to directly optimizing a search criteria.

6.1.2 Problem setup

Given a database $\mathcal{X} = \{x_1, \ldots, x_n\}$ and a query q, the search problem is to return top ranked items from the database that are most similar to the query. Let $s(q, x) \geq 0$ be a ranking function that measures the similarity between q and x. In large-scale search applications, the database size n can be billions or larger. Explicitly evaluating the ranking function $s(q, x)$ against all samples is very expensive. On the other hand, in order to achieve accurate search results, a complicated ranking function $s(q, x)$ is indispensible.

Modern search engines handle this problem by first employing a non-negative *selection function* $T(q, x)$ that selects a small set of candidates $\mathcal{X}_q = \{x : T(q, x) > 0, x \in \mathcal{X}\}$ with most of the top ranked items ($T(q, x) = 0$ means "not selected"). This is called **candidate selection** stage, which is followed by a **reranking** stage where a more costly ranking function $s(q, x)$ is evaluated on $\mathcal{X}_q$.

Two properties of the selection function $T(q, x)$ are: (1) It must be evaluated much more efficiently than the ranking function $s(q, x)$. In particular, for a given query, the complexity of evaluating $T(q, x)$ over the entire dataset should be sublinear or even constant, which is usually made possible by dedicated data structures such as inverted index tables. (2) The selection function is an approximation to $s(q, x)$. In other words, with high probability, the more similar q and x are, the

more likely x is contained in $\mathcal{X}_q$ (which means $T(q, x)$ should take a larger value).

This chapter focuses on the candidate selection stage, i.e., learning the selection function $T(q, x)$. In order to achieve both effectiveness and efficiency, three aspects need to be taken into account:

- $\mathcal{X}_q$ can be efficiently obtained (this is ensured by properties of selection function).
- The size of $\mathcal{X}_q$ should be small since it indicates the *computational cost* for reranking.
- The *retrieval quality* of $\mathcal{X}_q$ measured by the total similarity $\sum_{x \in \mathcal{X}_q} s(q, x)$ should be large.

Therefore, our objective is to retrieve a set of items that maximizes the ranking quality while lowering the computational cost (keeping the candidate set as small as possible). In addition, to achieve search efficiency, the selection stage employs the inverted index structure as in standard text search engines to handle web-scale dataset.

6.2 Learning to Search

This section presents the proposed *learning to search* framework. We present the general formulation first, followed by a specific algorithm based on boosted search trees.

6.2.1 Problem formulation

As stated in Section 6.1.2, the set of candidates returned for a query q is given by $\mathcal{X}_q = \{x \in \mathcal{X} : T(q, x) > 0\}$. Intuitively, the *quality* of this candidate set can be measured by the overall similarities while the reranking cost is linear in $|\mathcal{X}_q|$. Mathematically, we define:

$$\textbf{Retrieval Quality:} \qquad Q(T) = \sum_{q} \sum_{x \in \mathcal{X}} s(q, x) \mathbf{1}(T(q, x) > 0) \tag{6.1}$$

$$\textbf{Computational Cost:} \qquad C(T) = \sum_{q} \sum_{x \in \mathcal{X}} \mathbf{1}(T(q, x) > 0) \tag{6.2}$$

where $\mathbf{1}(\cdot)$ is the indicator function.

The **learning to search** framework considers the search problem as a machine learning problem that finds the optimal selection function T as follows:

$$\max_T Q(T) \quad \text{subject to } C(T) \leq C_0, \tag{6.3}$$

where C_0 is the upper-bound of computational cost. Alternatively, we can rewrite the optimization problem in (6.3) by applying Lagrange multiplier:

$$\max_T Q(T) - \lambda C(T), \tag{6.4}$$

where λ is a tuning parameter that balances the retrieval quality and computational cost.

To simplify the learning process, we assume that the queries are randomly drawn from the database. Let x_i and x_j be two arbitrary samples in the dataset, and let $s_{ij} = s(x_i, x_j) \in \{1, 0\}$ indicate if they are "similar" or "dissimilar". Problem in (6.4) becomes:

$$\begin{aligned} \max_T J(T) &= \max_T \sum_{i,j} s_{ij} \mathbf{1}(T(x_i, x_j) > 0) - \lambda \sum_{i,j} \mathbf{1}(T(x_i, x_j) > 0) \\ &= \max_T \sum_{i,j} z_{ij} \mathbf{1}(T(x_i, x_j) > 0) \end{aligned} \tag{6.5}$$

where

$$z_{ij} = \begin{cases} 1 - \lambda & \text{for similar pairs} \\ -\lambda & \text{for dissimilar pairs} \end{cases}. \tag{6.6}$$

6.2.2 Learning ensemble selection function via boosting

Note that (6.5) is nonconvex in T and thus is difficult to optimize. Inspired by AdaBoost [74], we employ the standard trick of using a convex relaxation, and in particular, we consider the exponential loss as a convex surrogate:

$$\min_T \mathcal{L}(T) = \sum_{i,j} e^{-z_{ij} T(x_i, x_j)} = \mathbb{E}[e^{-zT(x_i, x_j)}]. \tag{6.7}$$

Here we replace the summation over $\forall (x_i, x_j) \in \mathcal{X} \times \mathcal{X}$ by the expectation over two *i.i.d.* random variables x_i and x_j. We also drop the subscripts of z_{ij} and regard

z as a random variable conditioned on x_i and x_j.

We define the ensemble selection function as a weighted sum of a set of base selection functions:

$$T(x_i, x_j) = \sum_{m=1}^{M} c_m \cdot t_m(x_i, x_j). \tag{6.8}$$

Suppose we have learned M base functions, and we are about to learn the $(M+1)$-th selection function, denoted as $t(x_i, x_j)$ with weight given by c. The updated loss function is hence given by

$$\min_t \mathcal{L}(t, c) = \mathbb{E}[e^{-z[T(x_i,x_j)+ct(x_i,x_j)]}] = \mathbb{E}_w[e^{-czt(x_i,x_j)}], \tag{6.9}$$

where $\mathbb{E}_w[\cdot]$ denotes the *weighted expectation* with weights given by

$$w_{ij} = w(x_i, x_j) = e^{-z_{ij}T(x_i,x_j)} = \begin{cases} e^{-(1-\lambda)T(x_i,x_j)} & \text{for similar pairs} \\ e^{\lambda T(x_i,x_j)} & \text{for dissimilar pairs} \end{cases} \tag{6.10}$$

This reweighting scheme leads to the boosting algorithm in Algorithm 1.

In many application scenarios, each base selection function $t(x_i, x_j)$ takes only binary values 1 or 0. Thus, we may want to minimize $\mathcal{L}(t, c)$ by choosing the optimal value of $t(x_i, x_j)$ for any given pair (x_i, x_j).

Case 1: $t(x_i, x_j) = 0$

$$\mathcal{L}(t, c) = \mathbb{E}_w[e^{-0}] = 1. \tag{6.11}$$

Case 2: $t(x_i, x_j) = 1$

$$\mathcal{L}(t, c) = \mathbb{E}_w[e^{-zc}] = e^{-(1-\lambda)c} \cdot \mathrm{P}_w[s_{ij} = 1|x_i, x_j] + e^{\lambda c} \cdot \mathrm{P}_w[s_{ij} = 0|x_i, x_j]. \tag{6.12}$$

Comparing the two cases leads to:

$$t^*(x_i, x_j) = \begin{cases} 1 & \text{if } \mathrm{P}_w[s_{ij} = 1|x_i, x_j] > \frac{1-e^{-\lambda c}}{1-e^{-c}} \\ 0 & \text{otherwise} \end{cases} \tag{6.13}$$

To find the optimal c, we first decompose $\mathcal{L}$ in the following way:

$$\begin{aligned} \mathcal{L}(t, c) &= \mathbb{E}_w[e^{-czt(x_i,x_j)}] \\ &= \mathrm{P}_w[t(x_i, x_j) = 0|x_i, x_j] + e^{-c(1-\lambda)} \cdot \mathrm{P}_w[t(x_i, x_j) = 1, s_{ij} = 1|x_i, x_j] \\ &\quad + e^{c\lambda} \cdot \mathrm{P}_w[t(x_i, x_j) = 1, s_{ij} = 0|x_i, x_j]. \end{aligned} \tag{6.14}$$

Taking the derivative of $\mathcal{L}$ with respect to c, we arrive at the optimal solution for c:

$$c^* = \log \frac{(1-\lambda)\mathrm{P}_w[t(x_i,x_j)=1, s_{ij}=1|x_i,x_j]}{\lambda \mathrm{P}_w[t(x_i,x_j)=1, s_{ij}=0|x_i,x_j]}. \tag{6.15}$$

Algorithm 1 Boosted Selection Function Learning

Input: A set of data points $\mathcal{X}$; pairwise similarities $s_{ij} \in \{0,1\}$ and weights $w_{ij}=1$

1: **for** $m \in 1,2,\cdots,M$ **do**
2: Learn a base selection function $t_m(x,y)$ based on weights w_{ij}
3: Update ensemble: $T(x_i,x_j) \leftarrow T(x_i,x_j) + c_m \cdot t_m(x_i,x_j)$
4: Update weights: $w_{ij} \leftarrow w_{ij} \cdot e^{-z_{ij}c_m t_m(x_i,x_j)}$
5: **end for**

6.2.3 Tree implementation of the base selection function

Simultaneously solving (6.13) and (6.15) leads to the optimal solutions at each iteration of boosting. In practice, however, the optimality can hardly be achieved. This is particularly because the binary-valued base selection function $t(x_i,x_j)$ has to be selected from limited function families to ensure the wearability (finite model complexity) and more importantly, the efficiency. As mentioned in Section 6.1.2, evaluating $t(q,x)$ for $\forall x \in \mathcal{X}$ needs to be accomplished in sublinear or constant time when a query q comes. This suggests using an inverted table data structure as an efficient implantation of the selection function. Specifically, $t(x_i,x_j)=1$ if x_i and x_j get hashed into the same bucket of the inverted table, and 0 otherwise. This work considers trees (we name it "search trees") as an approximation to the optimal selection functions, and quick inverted table lookup follows naturally.

A natural consideration for the tree construction is that the tree must be balanced. However, we do not need to explicitly enforce this constraint: the balancedness is automatically favored by the term C in (6.4) as balanced trees give the minimum computational cost. In this sense, unlike other methods that explicitly enforce balancing constraint, we relax it while jointly optimizing the retrieval quality and computational cost.

Consider a search tree with L leaf nodes $\{\ell_1,\cdots,\ell_L\}$. The selection function

given by this tree is defined as

$$t(x_i, x_j) = \sum_{k=1}^{L} t(x_i, x_j; \ell_k), \tag{6.16}$$

where $t(x_i, x_j; \ell_k) \in \{0, 1\}$ indicating whether both x_i and x_j reach the same leaf node ℓ_k. Similar to (6.5), the objective function for a search tree can be written as:

$$\max_t J = \max_t \sum_{i,j} w_{ij} z_{ij} \sum_{k=1}^{L} t(x_i, x_j; \ell_k) = \max_t \sum_{k=1}^{L} J^k, \tag{6.17}$$

where $J^k = \sum_{ij} w_{ij} z_{ij} t(x_i, x_j; \ell_k)$ is a partial objective function for the k-th leaf node, and w_{ij} is given by (6.10).

The appealing additive property of the objective function J makes it trackable to analyze each split when the search tree grows. In particular, we split the k-th leaf node into two child nodes $k(1)$ and $k(2)$ if and only if it increases the overall objective function $J^{k(1)} + J^{k(2)} > J^k$. Moreover, we optimize each split by choosing the one that maximizes $J^{k(1)} + J^{k(2)}$.

To find the optimal split for a leaf node ℓ_k, we confine to the hyperplane split cases, i.e., a sample x is assigned to the left child $\ell_{k(1)}$ if $p^\top x + b = \tilde{p}^\top \tilde{x} > 0$ and right child otherwise, where $\tilde{p} = [p^\top \; b]^\top$ and $\tilde{x} = [x^\top \; 1]^\top$ are the augmented projection and data vectors. The splitting criterion is given by:

$$\begin{aligned} \max J^{k(1)} + J^{k(2)} &= \max_{\|\tilde{p}\|=1} \sum_{ij} w_{ij} z_{ij} \mathbf{1}(\tilde{p}^\top \tilde{x}_i \cdot \tilde{p}^\top \tilde{x}_j > 0) \\ &\approx \max_{\|\tilde{p}\|=1} \sum_{ij} w_{ij} z_{ij} [\tilde{p}^\top \tilde{x}_i \tilde{x}_j^\top \tilde{p}] \\ &= \max_{\|\tilde{p}\|=1} \tilde{p}^\top \tilde{X} M \tilde{X}^\top \tilde{p}, \end{aligned} \tag{6.18}$$

where $M_{ij} = w_{ij} z_{ij}$, and $\tilde{X}$ is the stack of all augmented samples at node ℓ_k. Note that as $\mathbf{1}(a > 0) = \frac{1}{2}\text{sign}(a) + \frac{1}{2}$ is non-differentiable, we approximate it using $\frac{1}{2}a + \frac{1}{2}$. The optimal $\tilde{p}$ of the above objective function is the eigenvector corresponding to the largest eigenvalue of $\tilde{X} M \tilde{X}^\top$.

The search tree construction algorithm is listed in Algorithm 2. In the implementation, if computation resource is critical, we may use stump functions to split the nodes with a large amount of samples, while applying the optimal projection p to the small nodes. The selection of the stump functions is similar to that in

traditional decision trees: on the given leaf node, a set of stump functions are attempted and the one that maximizes (6.17) is selected if the objective function increases.

Algorithm 2 Search Tree Construction

Input: A set of data points $\mathcal{X}$; pairwise similarities $s_{ij} \in \{0, 1\}$ and weights w_{ij} given by (6.10)

Output: Tree t

1: Assign $\mathcal{X}$ as root; enqueue root
2: **repeat**
3: Find a leaf node ℓ in the queue; dequeue ℓ
4: Find the optimal split for ℓ by solving (6.18)
5: **if** criteria in (6.17) increases **then**
6: Split ℓ into ℓ_1 and ℓ_2; enqueue ℓ_1 and ℓ_2
7: **end if**
8: **until** Queue is empty

6.2.4 Boosted search forest

In summary, we present a Boosted Search Forest (BSF) algorithm for the *learning to search* problem. In the learning stage, this algorithm follows the boosting framework described in Algorithm 1 to learn an ensemble of selection functions; each base selection function, in the form of a search tree, is learned with Algorithm 2. We then build inverted indices by passing all data points through the learned search trees. In analogy to text search, each leaf node corresponds to an "index word" in the vocabulary and the data points reaching this leaf node are the "documents" associated with this "index word". In the candidate selection stage, instead of exhaustively evaluating $T(q, x)$ for $\forall x \in \mathcal{X}$, we only need to traverse the search trees and retrieve all items that collide with the query example for at least one tree. The selected candidate set, given by $\mathcal{X}_q = \{x \in \mathcal{X} : T(q, x) > 0\}$, is statistically optimized to have a small size (small computation cost) while containing a large number of relevant samples (good retrieval quality).

6.3 Experiments

We evaluate the Boosted Search Forest (BSF) algorithm on several image search tasks. Although a more general similarity measure can be used, for simplicity we set $s(x_i, x_j) \in \{0, 1\}$ according to whether x_j is within the top K nearest neighbors (K-NN) of x_i on the designated metric space. We use $K = 100$ in the implementation.

We compare the performance of BSF to two most popular algorithms on high dimensional image search: k-means trees and LSH. We also compare to a representative method in the learning to hash community: spectral hashing, although this algorithm was designed for Hamming embedding instead of search. Here linear scan is adopted on top of spectral hashing for search, because its more efficient alternatives are either directly compared (such as LSH) or can easily fail as noticed in [72]. Our experiment shows that exhaustive linear scan is not scalable, especially with long hash codes needed for better retrieval accuracy (see Table 6.1).

The above algorithms are most representative. We do not compare with other algorithms for several reasons. Fist, LSH was reported to be superior to kd-trees [75] and spectral hashing was reported to out-perform RBM and BoostSCC [70]. Second, kd-trees and its extensions still work on low dimensions, and is known to behave poorly on high dimension data like in image search. Third, since this work focuses on learning to search, not learning to hash (Hamming embedding) or learning distance metrics that consider different goals, it is not essential to compare with more recent work on those topics such as [64, 65, 72, 66].

6.3.1 Concept-1000 dataset

This dataset consists of more than 150K images of 1000 concepts selected from the Large Scale Concept Ontology for Multimedia (LSCOM) [76]. The LSCOM categories were specifically selected for multimedia annotation and retrieval, and have been used in the TRECVID video retrieval series. These concept names were input as queries in Google and Bing, and the top returned images were collected.

We choose the image representation proposed in [1], which is a high dimensional ($\sim$84K) feature with reported state-of-the-art performance in many visual recognition tasks. PCA is applied to reduce the dimension to 1000. We then randomly select around 6000 images as queries, and use the remaining ($\sim$150K)

images as the search database.

In image search, we are interested in the overall quality of the set of candidate images returned by a search algorithm. This notion coincides with our formulation of the search problem in (6.4) that is aimed at maximizing retrieval quality while maintaining a relative low computational cost (for reranking stage). The number of returned images clearly reflects the computational cost, and the retrieval quality is measured by the recall of retrieved images, i.e., the number of retrieved images that are among the 100-NN of the query. Note that we use recall instead of accuracy because recall gives the upper-bound performance of the reranking stage.

Figure 6.1(a) shows the performance comparison with two search algorithms: k-means trees and LSH. Since our boosted search forest consists of tree ensembles, for a fair comparison, we also construct equivalent number of k-means trees (with random initializations) and multiple sets of LSH codes. Our proposed approach significantly outperforms k-means trees and LSH. The better performance is due to our learning to search formulation that simultaneously maximizes recall while minimizing the size of returned candidate set. In contrast, k-means trees uses only unsupervised clustering algorithm and LSH employs purely random projections. Moreover, the performance of the k-means algorithm deteriorates with the increasing dimensions.

It is still interesting to compare to spectral hashing, although it is not a search algorithm. Since our approach requires more trees when the number of returns increases, we implement spectral hashing with varying bits: 32-bit, 96-bit, and 200-bit. As illustrated in Figure 6.1(b), our approach significantly outperforms spectral hashing under all configurations. Although the search forest does not have an explicit concept of bits, we can measure it from the information theoretical point of view, by counting every binary-branching in the trees as one bit. In the experiment, our approach retrieves about 70% of 100-NN out of 500 returned images, after traversing 17 trees, each of 12 layers. This is equivalent to $17 \times 12 = 204$ bits. With the same number of bits, spectral hashing only achieves a recall rate around 60%.

6.3.2 One million tiny images

In order to examine the scalability of BSF, we conducted experiments on a much larger database. We randomly sample one million images from the 80 Millions Tiny Images dataset [77] as the search database, and 5000 additional images as queries. We use the 384-dimensional GIST feature provided by the authors of [77]. Comparison with search algorithms (Figure 6.2(a)) and hashing methods (Figure 6.2(b)) are made in a similar way as in the previous section. Again, the BSF algorithm substantially outperforms the other methods: using 60 trees (less than 800 bits), our approach retrieves 55.0% of the 100-NN with 5000 returns (0.5% of the entire database), while k-means trees achieves only 47.1% recall rate and LSH and spectral hashing are even worse. Note that using more bits in spectral hashing can even hurt performance on this dataset.

6.3.3 Search speed

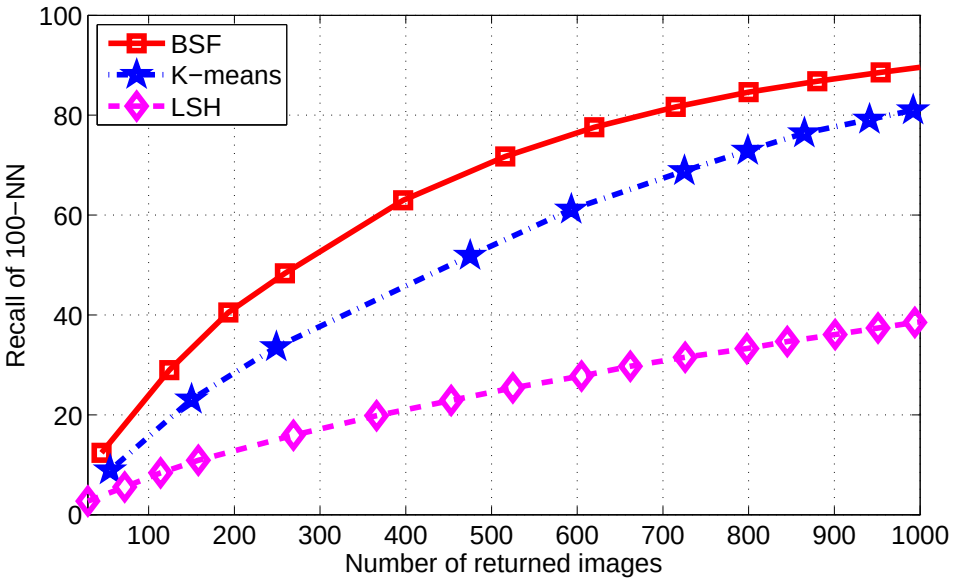
All three aforementioned search algorithms (boosted search trees, k-means trees, and LSH) can naturally utilize inverted index structures to facilitate very efficient search. In particular, both our boosted search trees and k-means trees use the leaf nodes as the keys to index a list of data points in the database, while LSH uses multiple independently generated bits to form the indexing key. In this sense, all three algorithms have the same order of efficiency (constant time complexity).

On the other hand, in order to perform search with compact Hamming codes generated by a learning to hash method (e.g. spectral hashing), one has to either use a linear scan approach or a hash table lookup technique that finds the samples within a radius-1 Hamming ball (or more complex methods like LSH). Although much more efficient, the hash table lookup approach is likely to fail as the dimension of hash code grows to a few dozens, as observed in [72]. The retrieval speed using exhaustive linear scan is, however, far from satisfactory. Table 6.1 clearly illustrates this phenomenon on a database of 0.5 billion synthesized items. Even small codes with 32 bits take around 1.55 seconds (without sorting). When the hash codes grow to 512 bits (which is not unusual for high-dimensional image/video data), the query time is almost 20 seconds. This is not acceptable for most real applications. On the contrary, our boosted search forest with 32 16-layer trees ($\sim$512 bits) responds in less than 0.073s. Our timing is carried out on a Intel Xeon Quad X5560 CPU, with a highly optimized implementation of Hamming

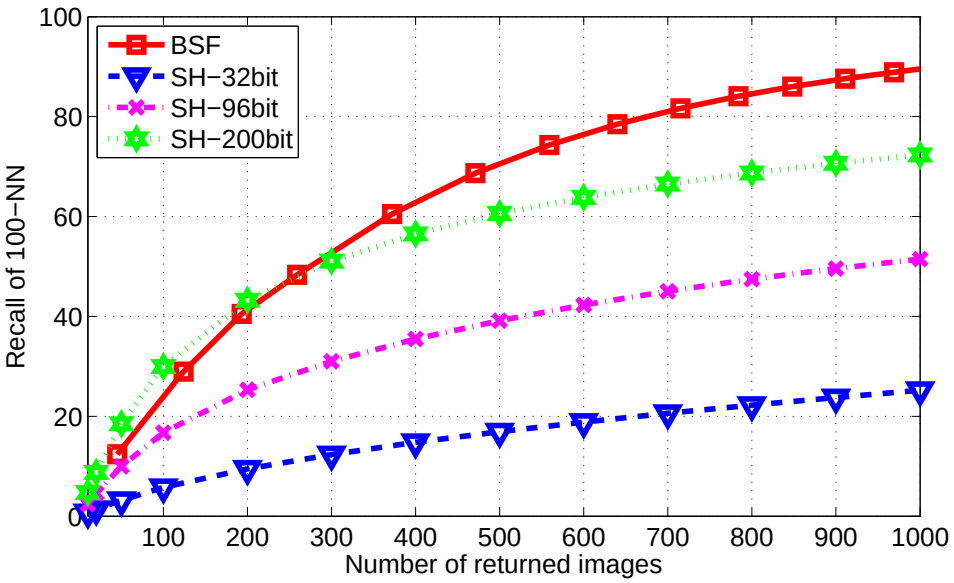
distance which is at least 8–10 times faster than a naive implementation.

Table 6.1: Comparison of retrieval time in a database with 0.5 billion synthesized samples.

#bits	**32**	**64**	**128**	**256**	**512**
Linear scan	1.55s	2.74s	5.13s	10.11s	19.79s
Boosted search forest	0.006s	0.009s	0.017s	0.034s	0.073s

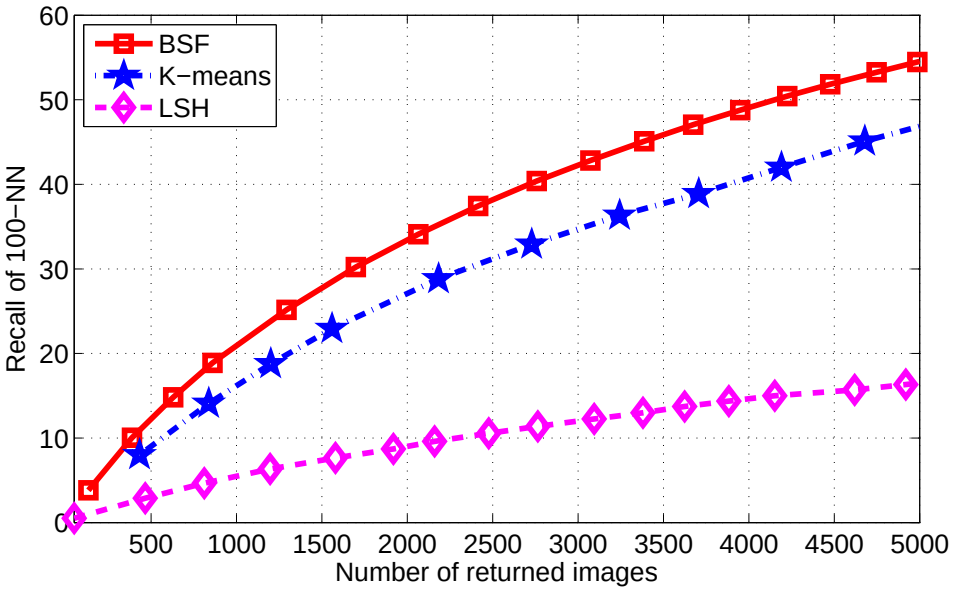


(a) Comparison of Boosted Search Forest (BSF) with k-means trees and LSH.

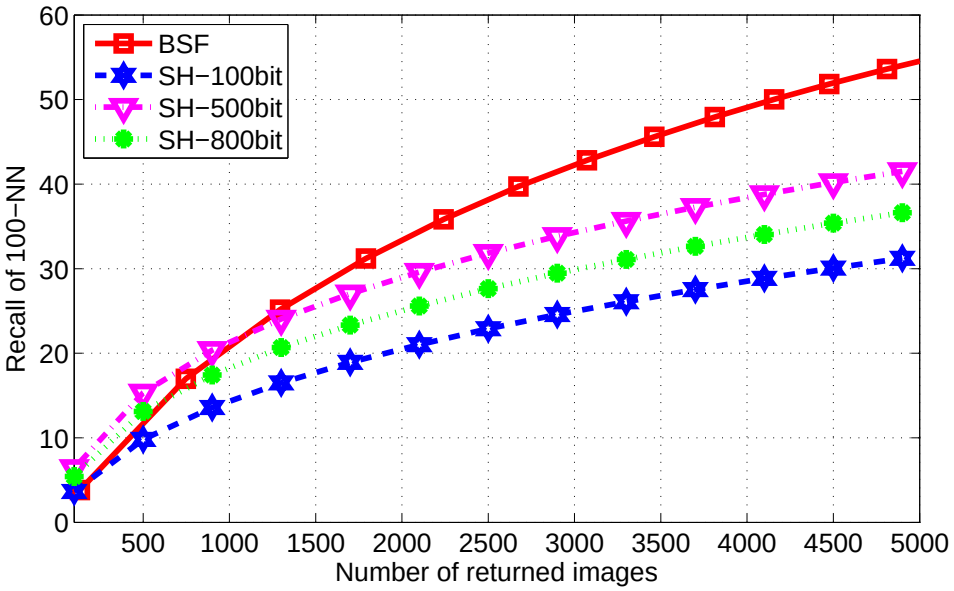


(b) Comparison of Boosted Search Forest (BSF) with Spectral Hashing (SH) of varying bits.

Figure 6.1: Experimental results on Concept-1000 dataset.



(a) Comparison of Boosted Search Forest (BSF) with k-means trees and LSH.



(b) Comparison of Boosted Search Forest (BSF) with Spectral Hashing (SH) of varying bits.

Figure 6.2: Experimental results on one million tiny images.

CHAPTER 7

CONCLUSION AND FUTURE WORK

In this dissertation we mainly study the person verification problem from the machine learning perspective. We are motivated by the two major challenges in practical person verification applications: limited sample issue and environment mismatch. To overcome these obstacles as well as the limitations of existing work, we propose several models, including both generative and discriminative ones. We also look into one potential application, person search in surveillance systems, and propose a machine learning framework for fast similarity search problem in a general sense. In this final chapter we briefly summarize our contributions and point out some directions for future research.

7.1 Contributions

The detailed contributions of this dissertation are listed as follows.

- We examine the two major challenges in person verification, limited sample issue and environment mismatch, in Chapter 2 and Chapter 5 respectively. The analysis of the these issues lays the foundation for the relationship between the conventional person identification problem, and the more fundamental and challenging problem – person verification. This also motivates the general methodologies for person verification, based on which we propose both generative and discriminative models.

- We propose the Covariance Verification Model (CVM) for person verification under a Bayesian framework. Under simple additive Gaussian assumptions we arrive at a verification model that bypasses the limited sample issue. Maximum a posteriori (MAP) estimation is employed for learning the model parameters. It incorporates prior knowledge and leads to more reliable parameter estimation, especially for small-scale datasets with a limited

number of people. In addition to standard EM algorithm, we also derive an approximate closed-form solution for CVM learning, which is much more efficient and proves to perform comparably. Experiments show that our approach consistently outperforms previous methods on both Multi-PIE and LFW datasets.

- We propose to learn a decision function for verification in a discriminative manner. This alleviates the need for any probabilistic distribution assumptions. The second-order formulation generalizes from both Bayesian Face Recognition (BFR) [2] and traditional metric learning (ML) approaches by offering a locally adaptive decision rule. We further formulate the inference on our decision function as a second-order large-margin regularization problem, and provide an efficient algorithm in its dual from. Compared with existing approaches including ML, our approach demonstrates state-of-the-art performance on several person verification benchmark datasets such as VIPeR, CAVIAR4REID, and LFW.

- We propose the Multi-setting Covariance Verification Model (MCVM) for person verification across settings. MCVM extends CVM by modeling the underlying correlations among different settings such that environment mismatch can be handled. It provides a unified model that accommodates all kinds of environmental variations, while many existing works only consider a special type (e.g. pose), or solve only the two-setting problems. Similar to CVM, we also provide efficient closed-form inferring algorithm for learning MCVM under the MAP criterion. Experimental results show significant improvement over universal CVM that treats all settings as the same.

- We introduce a learning-to-search framework for scalable similarity search in high dimensions. This lays the foundation for an efficient person search module in surveillance system. Unlike previous methods, our algorithm learns a boosted search forest by jointly optimizing search quality versus computational efficiency, under the supervision of pair-wise similarity labels. With a natural integration of the inverted index search structure, our method can handle web-scale datasets efficiently. Experiments show that our approach leads to better retrieval accuracy than the state-of-the-art search methods such as locality sensitive hashing and k-means trees.

7.2 Future Research

Models discussed in previous chapters seem to have resolved most issues in practical person verification applications. However, they are not without limitations. First, all the above methods consider only simple models, for instance, Gaussian assumptions in both CVM and MCVM, and second-order forms in the decision functions in Chapter 4. This inevitably limits the capability of the person verification models in dealing with complex situations, especially when huge intra-personal variations or environment mismatch exist. Second, while we focus on the machine learning aspects, an equally important problem, feature learning, has been largely neglected. Note that the features we use in our experiments are general image features such as Hierarchical Gaussianization [1], and not designed for face or human body images.

The recently popular deep learning (DL) techniques [78, 79] seem to be the remedy to both issues above. For one thing, they learn task-specific features directly from raw images. Thus they bridge feature learning and pattern classification (classification/verification) with a unified model. For the other, DL methods are usually implemented as deep neural networks (DNNs), which have the capacity of capturing arbitrarily complex models (given sufficiently many layers and large amount of training data).

Another direction that is of great interest to person verification problems is to take into account the domain knowledge. Specifically, we could make use of the well-studied 3D (pose) models or illumination models to eliminate or reduce the variations on these factors. Then machine learning would only need to focus on other variations, which makes the entire problem much easier.

REFERENCES

[1] X. Zhou, N. Cui, Z. Li, F. Liang, and T. S. Huang, "Hierarchical Gaussianization for image classification," in *IEEE International Conference on Computer Vision*, 2009, pp. 1971–1977.

[2] B. Moghaddam, T. Jebara, and A. Pentland, "Bayesian face recognition," *Pattern Recognition*, vol. 33, no. 11, pp. 1771–1782, 2000.

[3] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[4] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2nd ed. New York, NY: John Wiley and Sons, Inc., 2001.

[5] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[6] L. Yang and R. Jin, "Distance metric learning: A comprehensive survey," Michigan State Universiy, Tech. Rep., May 2006.

[7] M. Guillaumin, J. J. Verbeek, and C. Schmid, "Is that you? metric learning approaches for face identification," in *IEEE International Conference on Computer Vision*, 2009, pp. 498–505.

[8] H. V. Nguyen and L. Bai, "Cosine similarity metric learning for face verification," in *Asian Conference on Computer Vision*, 2010, pp. 709–720.

[9] M. Dikmen, E. Akbas, T. S. Huang, and N. Ahuja, "Pedestrian recognition with a learned metric," in *Asian Conference on Computer Vision*, 2010, pp. 501–512.

[10] E. Nowak and F. Jurie, "Learning visual similarity measures for comparing never seen objects," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[11] D. Gray and H. Tao, "Viewpoint invariant pedestrian recognition with an ensemble of localized features," in *European Conference on Computer Vision*, 2008, pp. 262–275.

[12] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar, "Attribute and simile classifiers for face verification," in *IEEE International Conference on Computer Vision*, 2009, pp. 365–372.

[13] Z. Cao, Q. Yin, X. Tang, and J. Sun, "Face recognition with learning-based descriptor," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2707–2714.

[14] T. Berg and P. N. Belhumeur, "Tom-vs-Pete classifiers and identity-preserving alignment for face verification," in *British Machine Vision Conference*, 2012, pp. 129.1–129.11.

[15] S. J. D. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *IEEE International Conference on Computer Vision*, 2007, pp. 1–8.

[16] Wikipedia, "Invertible matrix." [Online]. Available: http://en.wikipedia.org/wiki/Invertible_matrix

[17] Wikipedia, "Woodbury matrix identity." [Online]. Available: http://en.wikipedia.org/wiki/Woodbury_matrix_identity

[18] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 1–38, 1977.

[19] J. Gauvain and C. Lee, "Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 2, pp. 291–298, 1994.

[20] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, "Multi-PIE," *Image and Vision Computing*, vol. 28, no. 5, pp. 807–813, 2010.

[21] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," University of Massachusetts, Amherst, Tech. Rep. 07-49, October 2007.

[22] S. Yan, X. Zhou, M. Liu, M. Hasegawa-Johnson, and T. S. Huang, "Regression from patch-kernel," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.

[23] Z. Li, X. Zhou, and T. S. Huang, "Spatial gaussian mixture model for gender recognition," in *IEEE International Conference on Image Processing*, 2009, pp. 45–48.

[24] X. Zhou, X. Zhuang, S. Yan, S.-F. Chang, M. Hasegawa-Johnson, and T. S. Huang, "Sift-bag kernel for video event analysis," in *ACM International Conference on Multimedia*, 2008, pp. 229–238.

[25] D. G. Lowe, "Object recognition from local scale-invariant features," in *IEEE International Conference on Computer Vision*, 1999, pp. 1150–1157.

[26] G. Huang, M. Jones, E. Learned-Miller et al., "LFW results using a combined Nowak plus MERL recognizer," in *Workshop on Faces in Real-Life Images at European Conference on Computer Vision*, 2008.

[27] L. Wolf, T. Hassner, and Y. Taigman, "Similarity scores based on background samples," in *Asian Conference on Computer Vision*, 2009, pp. 88–97.

[28] Y. Ying and P. Li, "Distance metric learning with eigenvalue optimization," *Journal of Machine Learning Research*, vol. 13, pp. 1–26, 2012.

[29] J. Sánchez and F. Perronnin, "High-dimensional signature compression for large-scale image classification," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1665–1672.

[30] N. Gilardi and S. Bengio, "Local machine learning models for spatial data analysis," *Journal of Geographic Information and Decision Analysis*, vol. 4, no. 1, pp. 11–28, 2000.

[31] C. Burges, *Advances in kernel methods: support vector learning*. The MIT Press, 1999.

[32] S. Shalev-Shwartz, Y. Singer, and N. Srebro, "Pegasos: Primal estimated sub-gradient solver for SVM," in *International Conference on Machine Learning*, 2007, pp. 807–814.

[33] J. Platt, "Sequential minimal optimization: A fast algorithm for training support vector machines," Microsoft Research, Tech. Rep. 98-14, April 1998.

[34] R.-E. Fan, P.-H. Chen, and C.-J. Lin, "Working set selection using second order information for training support vector machines," *Journal of Machine Learning Research*, vol. 6, pp. 1889–1918, 2005.

[35] D. Cheng, M. Cristani, M. Stoppa, L. Bazzani, and V. Murino, "Custom pictorial structures for re-identification," in *British Machine Vision Conference*, pp. 68.1–68.11.

[36] D. Gray, S. Brennan, and H. Tao, "Evaluating appearance models for recognition, reacquisition, and tracking," in *Workshop on Performance Evaluation for Tracking and Surveillance at IEEE International Confenrece on Computer Vision*, 2007.

[37] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *Journal of Machine Learning Research*, vol. 10, pp. 207–244, 2009.

[38] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, "Information-theoretic metric learning," in *International Conference on Machine Learning*, 2007, pp. 209–216.

[39] M. Farenzena, L. Bazzani, A. Perina, V. Murino, and M. Cristani, "Person re-identification by symmetry-driven accumulation of local features," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2360–2367.

[40] K. Okuma, A. Taleghani, N. de Freitas, J. J. Little, and D. G. Lowe, "A boosted particle filter: Multitarget detection and tracking," in *European Conference on Computer Vision*, 2004, pp. 28–39.

[41] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[42] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 711–720, 1997.

[43] A. B. Ashraf, S. Lucey, and T. Chen, "Learning patch correspondences for improved viewpoint invariant face recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[44] A. Asthana, T. K. Marks, M. J. Jones, K. H. Tieu, and M. V. Rohith, "Fully automatic pose-invariant face recognition via 3d pose normalization," in *IEEE International Conference on Computer Vision*, 2011, pp. 937–944.

[45] T. Diethe, D. R. Hardoon, and J. Shawe-Taylor, "Multiview fisher discriminant analysis," in *Workshop on Learning from Multiple Sources at Neural Information Processing Systems Conference*, 2008.

[46] D. Lin and X. Tang, "Inter-modality face recognition," in *European Conference on Computer Vision*, 2006, pp. 13–26.

[47] A. A. Nielsen, "Multiset canonical correlations analysis and multispectral, truly multitemporal remote sensing data," *IEEE Transactions on Image Processing*, vol. 11, no. 3, pp. 293–305, 2002.

[48] A. Sharma, A. Kumar, H. D. III, and D. W. Jacobs, "Generalized multiview analysis: A discriminative latent space," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 2160–2167.

[49] M. Kan, S. Shan, H. Zhang, S. Lao, and X. Chen, "Multi-view discriminant analysis," in *European Conference on Computer Vision*, 2012, pp. 808–821.

[50] H. Hotelling, "Relations between two sets of variates," *Biometrika*, vol. 28, no. 3/4, pp. 321–377, 1936.

[51] R. Weber, H.-J. Schek, and S. Blott, "A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces," in *International Conference on Very Large Data Bases*, 1998, pp. 194–205.

[52] T.-Y. Liu, "Learning to rank for information retrieval," *Foundations and Trends in Information Retrieval*, vol. 3, no. 3, pp. 225–331, 2009.

[53] V. Jain and M. Varma, "Learning to re-rank: query-dependent image re-ranking using click data," in *International Conference on World Wide Web*, 2011, pp. 277–286.

[54] B. Kulis and T. Darrell, "Learning to hash with binary reconstructive embeddings," in *Advances in Neural Information Processing Systems*, 2009, pp. 1042–1050.

[55] D. Nistér and H. Stewénius, "Scalable recognition with a vocabulary tree," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2006, pp. 2161–2168.

[56] T. Yeh, J. J. Lee, and T. Darrell, "Adaptive vocabulary forests br dynamic indexing and category learning," in *IEEE International Conference on Computer Vision*, 2007, pp. 1–8.

[57] F. Moosmann, B. Triggs, and F. Jurie, "Fast discriminative visual codebooks using randomized clustering forests," in *Advances in Neural Information Processing Systems*, 2006, pp. 985–992.

[58] Y. Lin, F. Lv, S. Zhu, M. Yang, T. Cour, K. Yu, L. Cao, and T. S. Huang, "Large-scale image classification: Fast feature extraction and SVM training," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1689–1696.

[59] K. Fukunage and P. Narendra, "A branch and bound algorithm for computing k-nearest neighbors," *IEEE Transactions on Computers*, vol. 100, no. 7, pp. 750–753, 1975.

[60] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *International Conference on Computer Vision Theory and Applications*, 2009, pp. 331–340.

[61] J. S. Beis and D. G. Lowe, "Shape indexing using approximate nearest-neighbour search in high-dimensional spaces," in *IEEE Conference on Computer Vision and Pattern Recognition*, 1997, pp. 1000–1006.

[62] J. Friedman, J. Bentley, and R. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Transactions on Mathematical Software*, vol. 3, no. 3, pp. 209–226, 1977.

[63] C. Silpa-Anan and R. Hartley, "Optimised kd-trees for fast image descriptor matching," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[64] P. Jain, B. Kulis, and K. Grauman, "Fast image search for learned metrics," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[65] B. Kulis and K. Grauman, "Kernelized locality-sensitive hashing for scalable image search," in *IEEE International Conference on Computer Vision*, 2009.

[66] J. He, W. Liu, and S.-F. Chang, "Scalable similarity search with optimized kernel hashing," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2010, pp. 1129–1138.

[67] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *International Conference on Very Large Data Bases*, 1999, pp. 518–529.

[68] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Symposium on Computational Geometry*, 2004, pp. 253–262.

[69] G. Shakhnarovich, "Learning task-specific similarity," Ph.D. dissertation, Massachusetts Institute of Technology, 2005.

[70] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Advances in Neural Information Processing Systems*, 2008, pp. 1753–1760.

[71] R. Salakhutdinov and G. E. Hinton, "Semantic hashing," *International Journal of Approximate Reasoning*, vol. 50, no. 7, pp. 969–978, 2009.

[72] J. Wang, O. Kumar, and S.-F. Chang, "Semi-supervised hashing for scalable image retrieval," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[73] W. Liu, J. Wang, S. Kumar, and S.-F. Chang, "Hashing with graphs," in *International Conference on Machine Learning*, 2011, pp. 1–8.

[74] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: A statistical view of boosting," *The Annals of Statistics*, vol. 28, no. 2, pp. 337–374, 2000.

[75] G. Shakhnarovich, T. Darrell, and P. Indyk, *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice*. The MIT Press, 2006.

[76] M. Naphade, J. Smith, J. Tesic, S. Chang, W. Hsu, L. Kennedy, A. Hauptmann, and J. Curtis, "Large-scale concept ontology for multimedia," *IEEE Multimedia Magazine*, vol. 13, no. 3, pp. 86–91, 2006.

[77] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: A large data set for nonparametric object and scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 11, 2008.

[78] Q. V. Le, M. Ranzato, R. Monga, M. Devin, G. Corrado, K. Chen, J. Dean, and A. Y. Ng, "Building high-level features using large scale unsupervised learning," in *International Conference on Machine Learning*, 2012.

[79] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1106–1114.